Predicting time series of water levels using advanced AI models

KRZYSZTOF WIESNIAKOWSKI, University of Twente, The Netherlands

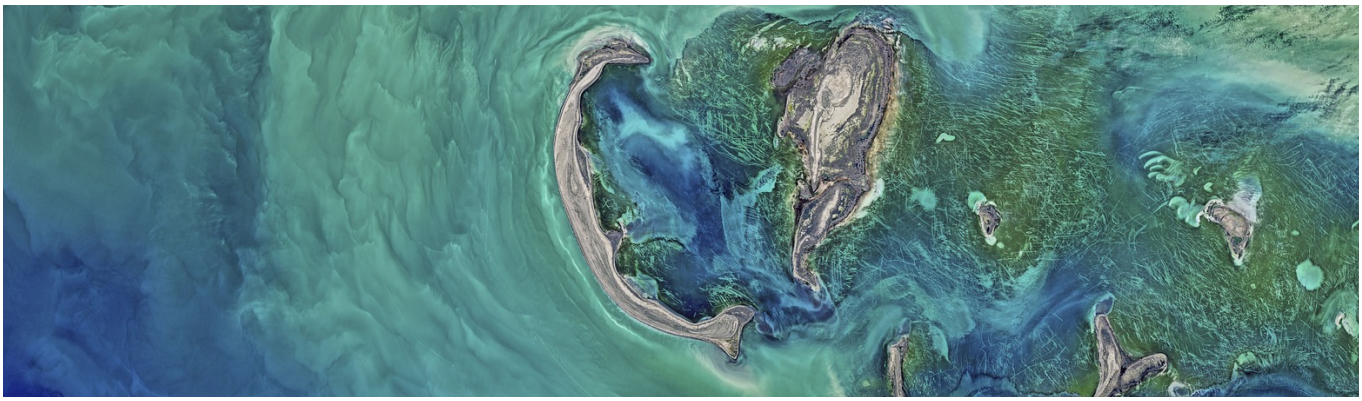


Fig. 1. Caspian Sea from a bird view [18]

ABSTRACT

Physics and mathematics allow us to provide accurate weather forecasts. Thanks to observations and numerical weather predictions, meteorologists can predict weather conditions which undoubtedly affect the daily choices made within various sectors of society. Models are built to provide forecasts to simulate atmospheric conditions and predict how they will evolve over time. By combining this information with hydrological data and models meteorologists can make predictions about water levels as well. However, various biases and errors are present in weather prediction models which are difficult to eliminate. We propose using machine learning as a post-processing technique to correct ocean circulation model outputs. In this research we carried out boosting algorithms together with a novel Fully Convolutional Network for the regression problem to see to what extent it is possible to successfully correct data coming from the ADCIRC model. We managed to obtain promising results which show that the model's outputs could be successfully corrected depending on the season of the year and on number of variables which were used to train the model.

Additional Key Words and Phrases: fully convolutional networks, post processing, time series forecasting, time series regression, water level prediction

1 INTRODUCTION

Accurate prediction of water levels in rivers and lakes is crucial for effective flood warning and management of water resources. As water level data from hydrological stations usually exhibit a time series structure, researchers often utilize time-series hydrological-prediction models to forecast future water-level data. By leveraging past data to predict future water levels, researchers can uncover hidden information and gain insight into future behavior. This insight is essential for mitigating flood effects, preventing disasters, and managing water resources effectively. At the moment weather prediction uses mathematical models of the atmosphere and water to predict the weather phenomenons. These models based on the physical and mathematical principles can predict short as well as long term weather forecasts [7]. However, weather models are prone to errors and biases which sometimes are very difficult to remove. As explained by the European Centre for Medium-Range Weather Forecasts ECMWF [16] in recent years there have been improvements in terms of optimizing weather models. Nevertheless, there are still persistent biases which are difficult to exclude. One of the models which is used for water level prediction is the ADvanced CIRCulation model ADCIRC [6]. This numerical ocean circulation model is used to simulate storm surge, tides, water levels and coastal circulation. However, this model in various situations lacks stability and as a consequence loses on its performance. Therefore, we are going to look at correcting this model output to achieve higher accuracy.

Water level data plays a vital role in water resource planning and management. Correcting the time series helps ensure that the data accurately represents the state of water bodies, such as rivers, lakes, and reservoirs. This information is essential for water allocation, assessing water availability, optimizing water use, and managing water resources sustainably. In this research, we try to predict the time series of water level using state-of-the-art regression techniques as well as with a deep learning architecture Fully Convolutional Networks FCN.

This research topic has been brought up by Infoplaza, a weather service forecast provider. The company has started a broader machine learning and artificial intelligence movement which should help the company to improve their weather forecasting. At the moment Infoplaza does not have a forecasting system that incorporates observations to predict future water levels. As a result, the company is interested in the improvement of water level forecasts in one location at the Caspian Sea. At this specific geographical location,

TScIT 39, July 7, 2023, Enschede, The Netherlands

it is necessary to apply corrections to the ADCIRC model to enhance its accuracy and provide Infoplaza with more dependable and trustworthy information.

2 RESEARCH QUESTION

RQ: *To what extent can advanced AI models predict time series of water levels in a closed reservoir?*

The following sub questions should help to answer this question

(1) *How do traditional models perform in predicting water levels in a closed reservoir compared to Fully Convolutional Network?*
(2) *Under what circumstances can the Machine Learning model be used and not be used ?*

3 RELATED WORK

In this section, we describe the state-of-the-art in predicting time series of water levels, we go over research work done towards numerical weather prediction and we take a closer look at the ADCIRC model, errors, and biases that concern weather prediction.

3.1 Correcting weather models

As it has been stated by Zadra et al. (2017) [25] weather models still need improvement due to differences between predictions and observations. These differences may reflect observational uncertainty, internal variability or errors and biases in the representation of physical processes.

Lee et al. (2021) [15] have carried out research in which peak storm surges were simulated. The ADCIRC model was tested on and it has been shown that due to its instability, 19 storms were not reported by the model. Thus, it can be noted that the aforementioned model can be improved.

In the field of machine learning and weather prediction, scientific studies demonstrate the successful utilization of these techniques to effectively mitigate errors and biases in weather models [10, 24]. In 2022, Atashi et al. [9] have applied a statistical method, machine learning algorithm as well as deep learning method to improve flood predictions in the Red River of the North [8]. After conducting their research on the particular case they have showed that Long Time Short–Memory LSTM has outperformed other methods used in the research.

This study aims to explore the effectiveness of deep neural networks and gradient boosting algorithms in addressing time series problems. Both approaches have demonstrated their success in overcoming the challenges inherent in time series analysis, and we seek to contribute to the existing knowledge by investigating their performance and suitability in this context.

3.2 Boosting algorithms in time series

Boosting algorithms have demonstrated practical applicability, particularly when employing decision trees as base models. The concept of "boosting" refers to a potent technique that adeptly combines multiple approximate and moderately accurate models to generate a highly precise predictive model. This is achieved through a series of iterations, where the learning algorithm repeatedly scans the training data and places emphasis on the examples that were previously misclassified. By giving more weight to these challenging examples, the algorithm combines the decisions from the weak rules to produce a final prediction with significantly improved accuracy [20].

Nguyen et al. (2021) [19] have developed hybrid models (GA-XGBoost and DE-XGBoost) combine a genetic algorithm GA and a differential evolution DE algorithm with the extreme gradient boosting XGBoost model to predict hourly water levels. They have used the which were collected from rain gauges and water level stations from 2003 to 2020, with 42 heavy rain events during that span. The comparison of results demonstrated the superior performance of two hybrid models, over classification and registration trees (CART) and random forest (RF) models in the multistep-ahead prediction of water levels.

Li et al. (2020) [17] in their work analysed two mobile phone activity dataset to predict the future traffic of mobile base stations in urban areas. They utilized gradient-boosted decision tree algorithm based on Kalman filter GBDT-KF together with LSTM model. They have concluded that the RMSE Root Mean Squared Error of the predicted values obtained from their GBDT-KF algorithm compared to the ground truth is only around 12-14% worse than that of the LSTM model. This points out that the GBDT-KF algorithm achieves a reasonable level of precision while significantly reducing training time by more than 100 times compared to the LSTM model. In essence, the GBDT-KF algorithm strikes a balance between accuracy and time complexity, making it a favorable alternative to the LSTM model.

3.3 Fully Convolutional Networks in time series

Wang et al. (2017) [23] in their research introduced Fully Convolutional Network FCN which showed promising results in time series. In their proposed architecture all layers are learnable which can provide high accuracy. To see the architecture please check Figure 5. They have tested this network architecture on 44 UCR time series datasets with other benchmarks [12]. It turned out that FCN has the lowest mean per-class error MPCE out of all the architectures which have been tested. In our research we will focus on time series regression. In this problem the objective is to make predictions about a continuously changing output variable over period of time, by considering its previous values and other factors that vary over time. Since FCN is relatively a new architecture, there is yet very little research done towards evaluating its performance. Nevertheless, Tan et al. (2021) [21] showed that FCN can be successfully applied to the regression problem as well. They have worked on their novel database archive for evaluating Time Series Extrinsic Regression TSER. An important aspect to recognize is that, they made an assumption that most recent values are not most indicative of future values and therefore, they did not evaluate the model on predicting for instance heart rate. They have concluded that FCN for most of the datasets and for multivariate time series performs better than other traditional machine learning algorithms.

4 METHODOLOGIES

4.1 Literature review and business understanding

Gradient boosting algorithms have shown encouraging results in predictive capabilities in terms of accuracy metrics and training

speed [11]. Moreover, aforementioned algorithms showed promising results for time series problems as well. Therefore, these algorithms can be used as a great benchmark for FCN for a regression problem.

Although there has been research done in the field of using machine learning techniques to improve numerical weather predictions, there is still a gap for more research when it comes to using deep learning methods for univariate time series regression problems in water level predictions. After researching in which fields FCN has been applied and evaluated, it can be concluded that there is very little research when it comes to FCN usage in numerical weather forecasting such as correcting water level forecasts.

This paper will contribute in analyzing the traditional ML techniques together with FCN architecture to see how deep learning performs compared to the state-of-the-art techniques for predicting time series regression problems.

Furthermore, this research aims to improve water level prediction produced by the ADCIRC model by using machine learning as a post-processing technique. This approach is expected to enhance the accuracy of services provided by Infoplaza by providing a more accurate time series of water levels in one location at the Caspian Sea.

4.2 Implementation

In this research, we utilize traditional learning regression techniques as well as a Fully Convolutional Network. This deep learning method has been implemented using Tensforflow [5] and Keras [22] in Python language. To see how traditional regression techniques perform, we used PyCaret [4] since it enabled us to rapidly prototype, compare, and evaluate machine learning models using a user-oriented and intelligible interface. The exact results of using Pycaret are described in an Appendix B.1 section.

4.3 Machine Learning Architecture

Fully Convolutional Network for a regression problem has been inspired and adapted from previous research done by Wang et al. (2017) [23]. In their research authors dealt with the classification problem. They constructed the ultimate networks by combining three convolution blocks with filter sizes of 128, 256, 128 in each block. Following the convolution blocks, the features are passed through a global average pooling layer, instead of a fully connected layer, which significantly decreases the number of weights. In the end, a softmax layer generates the ultimate label. However, in our problem we carry out a regression problem, therefore, an architecture needs to be adapted.

Tan et al. (2021) [21] have successfully applied FCN to the regression problem. In their research global average pooling and softmax activation function which normally are used for classification problems were replaced with 1 dimensional global average pooling and linear activation function which allowed the network to predict continuous values.

In our research, we utilize the same network architecture which can be seen in the figure 2. The linear function is defined as:

$$f(x) = mx + b \tag{1}$$

where m is the slope and b is the y-intercept.

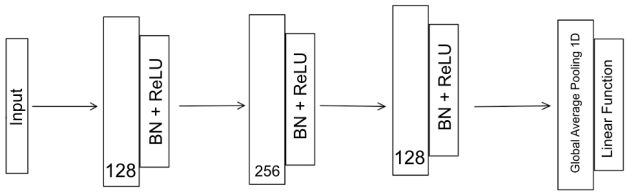


Fig. 2. FCN architecture for regression

5 EXPERIMENTAL SETUP

5.1 Dataset format

The dataset consists of data generated with the ADCIRC model using ECMWF atmospheric forcing data [1] as well as observations from NCOC [14]. Model outputs have been generated by Infoplaza employees for the time span of one year at the same location at Caspian Sea. Data was produced for one year starting from May 2022 up to and including April 2023.

ADCIRC model outputs are in JSON format consists of several information: what time model has been run, predictions for 10 days in advance with 30 min timestamp about water levels, wind speed and wind directions. ADCIRC model is run twice per day. 720 JSON files were used in this research.

NCOC observations are generated for every month in a CSV format with 10 minute timestamps and the following information: water level, water temperature and conductivity. This research utilized a total of 12 CSV files.

Water levels are expressed as the delta value in meters compared to the reference level specified which is based on bathymetry data used inside Infoplaza.

5.2 Data cleaning

To make sure that the data we used in our research is consistent, complete, and in suitable format we made sure that observations as well as model outputs which were used in the training process are free of null, extraordinary high, or low values which could be for example a result from sensors' malfunctions. Therefore, we excluded all the observations that either were null or had differences between consecutive values greater than 0.5 meters.

5.3 Data preprocessing

To use both inputs as well as observations in our experiment they were brought to the same format and merged based on the timestamp. Following this, 12 pandas DataFrame [3] objects were obtained with the information from observations as well as ADCIRC model outputs.

The next step was to create a file that could be used in the experiments. Since the goal of the research is to use machine learning as a post-processing technique, for every prediction we used 9 ADCIRC model outputs as the inputs for our ML models. Therefore, files were created with the following information: observation time, forecast runtime, forecast 8 timestamps ago, forecast 7 timestamps ago, ..., forecast 1 timestamp ago, forecast at that time, and observation at that time.

Since the ADCIRC model produces predictions 10 days in advance, experiments also take into consideration to what extent can machine learning be used depending on how long in the future model outputs are corrected. Therefore, based on model runtime and 3 lead times; 0-12 hours, 12-48 hours, and 48-240 hours we included model predictions together with the upper value of model runtime to the dataset. That means that for every model runtime model predictions and observations were appended together with an upper range time. Our dataset was saved in a CSV as well as numpy format [2] which were used in experiments.

That led to a file for each month with the following information: upper limit lead time, forecast 8 timestamps ago, forecast 7 timestamps ago, ..., forecast 1 timestamp ago, forecast at that time, and observation as a target column. In the last stage, every file which corresponds to one month was divided into training, validation, and testing sets with the proportions 80 %, 10%, 10 %, accordingly. Splitting the data from every month into these 3 sets allowed us to train the models for the whole year and later evaluate them for the whole year as well as for a specific season without the need of retraining them again.

In the process of preparing the data for analysis, 86330 rows have been excluded before appending them to the dataset used in experiments. This is because they lacked consecutive values for the window size and could not be used inside datasets. This could be the result of missing model predictions or observations for certain timestamps. Eventually, the whole dataset consists of 212886 rows for training data, 26601 rows for validation data, 26643 rows for testing data.

5.4 Experiments

Since our goal is to successfully correct the ADCIRC model outputs we established its forecasts as the baseline for our results. Moreover, we analyze to what extent we can correct the outputs of ADCIRC model depending on the season of the year. All the models have been trained for 1 year and after the training process they were evaluated for the whole year as well as every season with the testing data which was not used in the training process.

Before training, data was normalized with the following formula:

$$x_{\text{normalized}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{2}$$

This linearly transformed data to fit the interval [0,1].

Normalizing data before training ensured features were on a similar scale, avoiding domination, aiding convergence, enabling equitable comparisons, reducing sensitivity to initial conditions, and facilitating regularization.

The maximum value from the dataset used in experiments was 240 since we appended the upper range model lead times, and the minimum value was -2.2379 which represents the lowest water level inside data collection.

5.5 Evaluation metrics

To assess a performance from different models we evaluate them using the following metrics. Where n is the total number of data points, y is the actual (true) value of the target variable for the i-the data point, $\hat{y}$ is the predicted value of the target variable for the i-th data point.

The Mean Absolute Error MAE measures the average absolute difference between the predicted values and the actual values in a dataset. It provides a direct interpretation of the average magnitude of errors. A lower MAE indicates better performance, as it means the model has smaller average errors.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{D}|y_i - \hat{y}_i| \tag{3}$$

The Mean Squared Error MSE measures the average squared difference between the predicted values and the actual values. It squares the errors, which means it penalizes larger errors more heavily. A lower MSE indicates better performance, as it means the model has smaller overall errors.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4}$$

Root Mean Squared Error RMSE

$$\text{RMSE} = \sqrt{\frac{\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2}{N}} \tag{5}$$

6 RESULTS AND DISCUSSION

All the values inside the tables are in centimetres and present trained model predictions on testing data together with model ADCIRC outputs for the same dates.

6.1 Univariate time series

Firstly, the experiments were run using only water levels as the inputs. In this set up, we wanted to observe whether exclusively water levels are enough to successfully correct ADCIRC model outputs. After running an experiment with Pycaret [4] with water level data solely, it has shown that boosting algorithms have a great potential to improve time series with the aforementioned data.

The results for univariate time series experiments are presented in the table 1. ADCIRC model has lower MSE for the whole year comparing to boosting regressors algorithms and FCN. However, post-processed predictions have marginally lower MAE compared to ADCIRC and FCN. The reason for that is that RMSE amplifies the impact of larger errors due to the squaring operation. A lower RMSE indicates that the model has smaller overall errors, particularly with regard to larger errors or outliers. On the other hand, MAE treats all errors equally, regardless of their magnitude. That means boosting algorithms predictions that are generally closer to the true values than ADCIRC predictions. However, as we can see in Figure 3, they have a few predictions that are significantly off, contributing to a higher RMSE.

Table 1. Metrics for the whole year of testing data for univariate time series

Model	MSE	RMSE	MAE
ADCIRC	127.73	11.30	8.63
Cat Boost	194.96	13.96	8.58
Gradient Boosting	196.15	14.01	8.61
Light Gradient Boosting	196.10	14.00	8.57
FCN	194.82	13.96	8.74

After providing evaluation metrics for the whole year, the next step was to analyze how machine learning models perform in specific seasons. Therefore, we have also evaluated them in summer where we included June, July, and August 2022. For autumn we have included September, October, and November 2022, for winter December 2022, January, and February 2023, for spring March, and April 2023.

Boosting algorithms as well as FCN outperform ADCIRC for most of the year except autumn time where they perform significantly worse compared to our baseline. Moreover, also ADCIRC performs slightly worse in this particular season compared to the rest of the year. More extensive research is needed to see whether machine learning models are able to successfully correct ADCIRC model outputs in this particular season.

6.2 Multivariate time series

The next experiments were run including multiple variables such as wind u, wind v, and wind speed to see whether the models' performance can be improved. The results are presented in the tables below and discussed later in the section. PyCaret [4] was one more time run to see whether different models present significantly better than boosting algorithms. The results present greater differences between each other compared to running PyCaret for univariate time series. Despite the fact that boosting algorithms were already assessed for univariate problems, they will be re-evaluated in the multivariate section of this research to provide a comprehensive overview.

Table 4. Metrics for the whole year of testing data for multivariate time series

Model	MSE	RMSE	MAE
ADCIRC	127.73	11.30	8.63
Cat Boost	174.42	13.21	8.19
Gradient Boosting	179.05	13.38	8.34
Light Gradient Boosting	176.62	13.29	8.20
FCN	226.50	15.05	10.31

After running experiments for multivariate time series it can be observed that there is a slight improvement in terms of validation metrics for boosting algorithms. They improved in all criteria, MSE, RMSE, and MAE in the scope of the whole year. After deeper analysis, we can see that there is an improvement in winter, and autumn and for some of the algorithms in spring table 3. They still show diminished results than ADCIRC during autumn. Nevertheless, there is a marginal improvement compared to the model which used only water levels as inputs.

On the other hand, FCN dropped its performance in every season for the whole year compared to univariate time series. A possible reason for that may be overfitting where the model may have memorized the patterns and specific details of the training data without learning to generalize well to unseen examples. Another possible reason could be that this model architecture may not be well suited for such a long sequence input length. Therefore, a more comprehensive analysis is needed to understand this phenomenon. Several concepts are described in detail in the Future work section 8.

6.3 Lead time

Since these algorithms were used as a post-processing technique it is vital to look at to what extent these algorithms correct outputs depending on how long in the future ADCIRC model predicts. As it has been already described in the Data Preprocessing 5.3 section, every row in the dataset consists of the upper time range in which the model predictions were produced. That allowed us to filter the data, based on the time horizon it was generated and analyze to what extent ADCIRC model outputs are corrected depending on the time interval.

Both tables show MSE, RMSE, and MAE for the specific lead times in the scope of the whole year for predictions on the testing data. Table 5 and Table 6 present that the ADCIRC model in all the lead times has lower RMSE compared to boosting algorithms as well as to FCN. In contrast, boosting algorithms have lower MAE almost in every lead time compared to the ocean circulation model. We can see that univariate time series perform better compared to multidimensional when it comes to predicting water levels closer in the future whereas multivariate has lower MAE when looking at longer time horizon.

Table 5. Metrics for the specific lead times for the whole year of testing data for univariate time series

Metric	MSE			RMSE			MAE		
Model	0-12h	12-48h	48-240h	0-12h	12-48h	48-240h	0-12h	12-48h	48-240h
ADCIRC	19.89	55.50	153.26	4.46	7.45	12.38	3.64	5.92	9.66
Cat Boost	24.60	63.20	239.32	4.96	7.95	15.47	3.55	5.47	9.66
Gradient Boosting	26.72	64.48	240.25	5.17	8.03	15.50	3.64	5.50	9.73
Light Gradient Boosting	27.46	64.80	229.52	5.24	8.05	15.49	3.61	5.51	9.68
FCN	37.45	83.54	253.13	6.12	9.14	15.91	4.35	6.18	9.95

Table 6. Metrics for the specific lead times for the whole year of testing data for multivariate time series

Metric	MSE			RMSE			MAE		
Model	0-12h	12-48h	48-240h	0-12h	12-48h	48-240h	0-12h	12-48h	48-240h
ADCIRC	19.89	55.50	153.26	4.46	7.45	12.38	3.64	5.92	9.66
Cat Boost	48.86	81.00	206.21	6.99	9.00	14.36	4.02	5.78	9.08
Gradient Boosting	35.76	70.56	215.50	5.98	8.40	14.68	3.88	5.62	9.33
Light Gradient Boosting	47.75	77.09	210.25	6.91	8.78	14.50	4.25	5.77	9.07
FCN	64.64	84.45	243.67	8.04	9.19	15.61	5.92	6.87	10.41

6.4 Experiments with non-boosting algorithms

PyCaret [4] enabled us to conduct experiments involving a wider variety of models, going beyond just boosting algorithms. Nevertheless, to narrow down the focus of this research and to make an extensive analysis of boosting algorithms' application on correcting ADCIRC model, other machine learning models were not analyzed

Table 2. Metrics for the specific seasons for testing data for univariate time series

Season	Summer			Autumn			Winter			Spring		
Model	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE
ADCIRC	115.77	10.76	8.02	220.79	14.85	12.02	51.90	7.20	6.28	126.22	11.23	7.95
Cat Boost	88.29	9.40	7.00	594.28	24.38	17.72	22.20	4.71	3.35	84.72	9.20	5.81
Gradient Boosting	89.14	9.44	7.08	597.23	24.43	17.77	19.10	4.37	3.36	84.90	9.21	5.77
Light Gradient Boosting	88.42	9.40	7.03	603.56	24.57	17.78	22.10	4.70	3.36	79.00	8.89	5.69
FCN	96.53	9.82	7.56	583.01	24.15	17.64	21.25	4.61	3.41	83.93	9.23	5.95

Table 3. Metrics for the specific seasons for testing data for multivariate time series

Season	Summer			Autumn			Winter			Spring		
Model	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE
ADCIRC	115.77	10.76	8.02	220.79	14.85	12.02	51.90	7.20	6.28	126.22	11.23	7.95
Cat Boost	96.62	9.83	7.22	500.91	22.57	16.26	16.71	4.10	3.02	82.57	9.08	5.83
Gradient Boosting	89.43	9.45	7.16	541.89	23.27	17.05	18.39	4.29	3.21	71.18	8.44	5.41
Light Gradient Boosting	100.57	10.02	7.27	514.42	22.68	16.23	17.03	4.13	3.06	83.77	9.15	5.88
FCN	118.79	10.76	8.01	582.49	24.15	17.60	106.52	10.32	8.97	124.76	11.16	8.95

to such a broad extent. To see the metrics after running PyCaret please see Figure 6 and 7.

6.5 Visualizations

Figure 3 shows visualizations for the selected months where ADCIRC model outputs are plotted together with a chosen boosting algorithm, FCN, and NCOC observations. Looking at the graphs it affirms what can be observed in the tables 2 and 3. We can examine on the graph that indeed post-processed ADCIRC model outputs are successful in spring since boosting algorithm follow observations much closer than the original ADCIRC model. Nevertheless, boosting algorithms as well as FCN fail to near NCOC observations during autumn.

6.6 Discussion

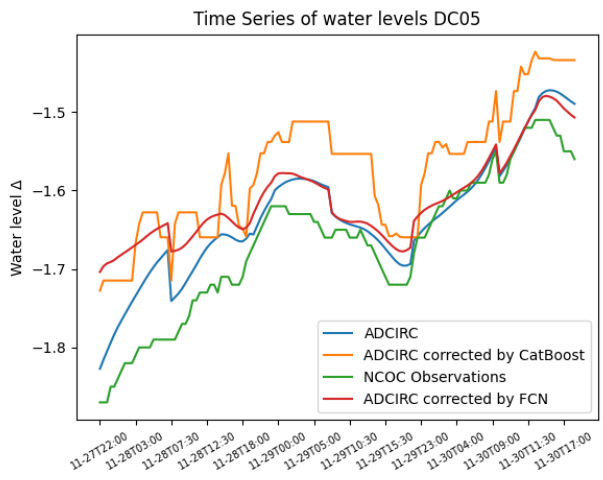
After thorough research, we have obtained results relevant to our primary research question.

(1) Our first sub-research question *How do traditional models perform in predicting water level in a closed reservoir compared to Fully Convolutional Network?* is depicted in the tables in the sections 6.1, 6.2 and in 6.5 sections where evaluation metrics, as well as visualizations, are shown for both boosting algorithms and FCN.
(2) To answer our second sub-research question *Under what circumstances can the Machine Learning model be used and not be used ?* we started by first runnning experiments with PyCaret [4] to choose suitable algorithms for this regression problem. We observed that boosting algorithms show promising results. Secondly, after thoroughly analyzing the performance of the models in each of the seasons both for univariate and multivariate we can conclude that using Machine Learning model can be used when it has lower RMSE as well as MAE compared to the ADCIRC model. Only then the corrected outputs are closer to the actual observations since they have smaller significant errors and perform better on average for most of the data points.

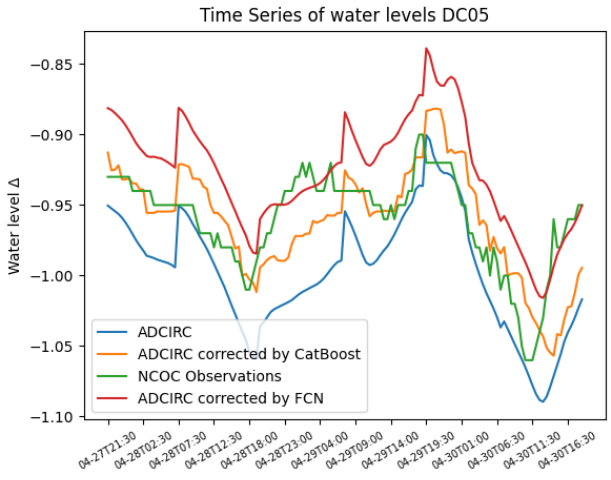
As has been shown in the experiments, FCN has a better performance in univariate time series compared to multidimensional. The possible cause for that may be that when dealing with shorter input sequences, the receptive field of the convolutional layers can cover a larger portion of the sequence, capturing more context and capturing patterns effectively. This is because the information from the input sequence has fewer steps to propagate through the convolutional layers, allowing the network to consider more comprehensive context information within a shorter distance. On the other hand, when processing longer input sequences, the receptive field of the convolutional layers might not cover the entire context required for accurate predictions. The limited receptive field means that the network may only consider a portion of the context, potentially missing out on important patterns or long-term dependencies in the data of the context. In this research all variables were used in the multivariate experiments. That significantly extended the length of the input sequence for the FCN network and as it has been already explained convolutional layers may have not captured all the important patterns or long-term dependencies in the data.

7 CONCLUSION

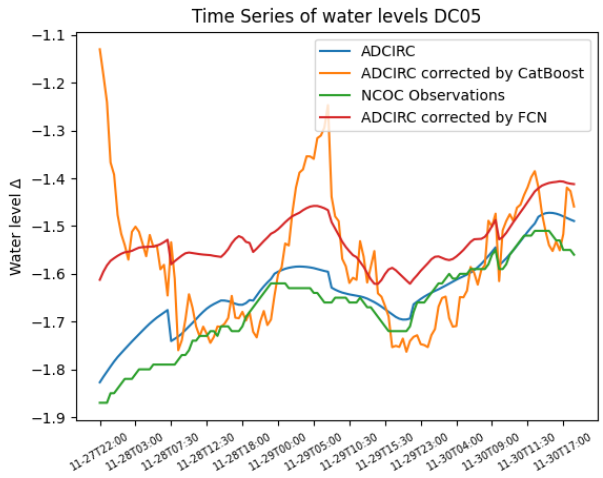
In conclusion, we have proven in our research that ML can be used to successfully correct ADCIRC model outputs depending on the season of the year. Overall, for the whole year, the ADCIRC model still has a lower RMSE than other machine learning models used in this research, which indicates that has smaller overall errors and fewer outliers. Nevertheless, boosting algorithms show better average precision for both univariate and multivariate compared to ADCIRC. FCN still shows relatively good performance when it comes to univariate times series but significantly dropped when other variables were included in the experiments.



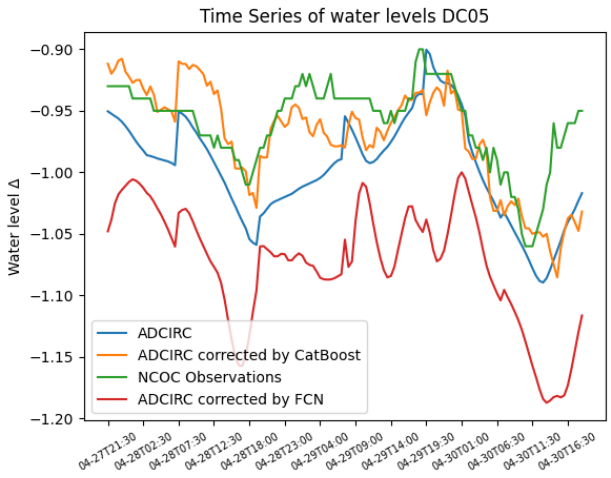
(a) November 2022, ADCIRC post processed by CatBoost Regressor and FCN for univariate time series



(b) April 2023, ADCIRC post processed by CatBoost Regressor and FCN for univariate time series



(c) November 2022, ADCIRC post processed by CatBoost Regressor and FCN for mutlivariate time series



(d) April 2023, ADCIRC post processed by CatBoost Regressor and FCN for multivariate time series

Fig. 3. Graphs presenting ADCIRC post processed by CatBoost Regressor and FCN for November 2022 as well as April 2023 both for univariate and multivariate time series

8 FUTURE WORK

This research has shown that boosting algorithms and FCN for univariate time series may be very beneficial in correcting ADCIRC outputs. Nevertheless, future work is needed to further investigate whether it is possible to exceed current results.

In this research, only a window of 9 timestamps was used for both univariate and multivariate time series. In future research experimenting with different window sizes may bring promising results.

Furthermore, within this study, all lead times were included in the training process. Possibly in the future, the models could be trained only in specific lead times to see whether the models perform remarkably well only in specific time horizons.

Another factor that could allow to understand the data more in-depth and enhance the results could be Principal Component Analysis PCA when running experiments with multivariate time series. PCA would allow to simplify and understand multidimensional data. It would identify the most important directions of variation in the data and project the data in these directions. PCA requires data to be standardized to remove differences in scales and units. It

then calculates the covariance matrix to capture the relationships between the features. Next, it performs an eigenvalue-eigenvector decomposition on the covariance matrix, where the eigenvectors represent the directions of maximum variance.

The eigenvectors, known as principal components PCs, are ranked based on their eigenvalues, indicating the amount of variance they explain. By selecting a subset of the PCs, it is possible to reduce the dimensionality of the data while retaining most of the important information. This could be very beneficial for improving the results for multivariate time series specifically for experiments which include FCN since it would reduce the data dimensionality and would enable convolutional layers to capture all important patterns.

9 ACKNOWLEDGEMENTS

I would like to thank Christian Versloot who was always eager to help me, provided valuable pieces of advice both from data science as well as from industry perspective, and guided me throughout the process. I would like to also thank my supervisor Elena Mocanu who constantly challenged me to improve my research.

REFERENCES

[1] [n. d.]. ECMWF. https://www.ecmwf.int/en/forecasts/datasets/search
[2] [n. d.]. numpy.array - NumPy v1.24 Manual. https://numpy.org/doc/stable/reference/generated/numpy.array.html?highlight=array
[3] [n. d.]. pandas.DataFrame - pandas 2.0.1 documentation. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html
[4] [n. d.]. PyCaret 3.0 - Docs. https://pycaret.gitbook.io/docs/
[5] [n. d.]. Tensorflow API docs. https://www.tensorflow.org/api_docs/python/tf
[6] 2023. ADCIRC. https://en.wikipedia.org/wiki/ADCIRC
[7] 2023. Numerical weather prediction. https://en.wikipedia.org/wiki/Numerical_weather_prediction
[8] 2023. Red river of the north. https://en.wikipedia.org/wiki/Red_River_of_the_North
[9] Vida Atashi, Hamed Taheri Gorji, Seyed Mojtaba Shahabi, Ramtin Kardan, and Yeo Howe Lim. 2022. Water level forecasting using Deep Learning Time-series analysis: A case study of red river of the north. *Water* 14, 12 (2022), 1971. https://doi.org/10.3390/w14121971
[10] Massimo Bonavita and Patrick Laloyaux. 2020. Machine learning for model error inference and correction. *Journal of Advances in Modeling Earth Systems* 12, 12 (2020). https://doi.org/10.1029/2020ms002232
[11] Debaditya Chakraborty, Hosam Elhegazy, Hazem Elzarka, and Lilianna Gutierrez. 2020. A novel construction cost prediction model using hybrid natural and light gradient boosting. *Advanced Engineering Informatics* 46 (2020), 101201. https://doi.org/10.1016/j.aei.2020.101201
[12] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. 2015. The UCR Time Series Classification Archive. www.cs.ucr.edu/~eamonn/time_series_data/.
[13] Margarita Granat. 2019. How to use Convolutional Neural Networks for time series classification. https://towardsdatascience.com/how-to-use-convolutional-neural-networks-for-time-series-classification-56b1b0a07a57
[14] Azamat Ismagulov. [n. d.]. About north caspian project. https://www.ncoc.kz/en/ncoc/about
[15] Jun-Whan Lee, Jennifer L. Irish, Michelle T. Bensi, and Douglas C. Marcy. 2021. Rapid prediction of peak storm surge from tropical cyclone track time series using Machine Learning. *Coastal Engineering* 170 (2021), 104024. https://doi.org/10.1016/j.coastaleng.2021.104024
[16] Jose Andres Perez Leon. 2022. Addressing biases in near-surface forecasts. https://www.ecmwf.int/en/newsletter/157/meteorology/addressing-biases-near-surface-forecasts
[17] Ling Li, Sida Dai, Zhiwei Cao, Jinghui Hong, Shu Jiang, and Kunmeng Yang. 2020. Using improved gradient-boosted decision tree algorithm based on Kalman filter (GBDT-KF) in time series prediction. *The Journal of Supercomputing* 76, 9 (2020), 6887–6900. https://doi.org/10.1007/s11227-019-03130-y
[18] NASA. 2016. *Aerial photograph of islands.* https://unsplash.com/photos/WKT3TE5AQu0
[19] Duc Hai Nguyen, Xuan Hien Le, Jae-Yeong Heo, and Deg-Hyo Bae. 2021. Development of an Extreme Gradient Boosting Model Integrated With Evolutionary Algorithms for Hourly Water Level Prediction. *IEEE Access* 9 (2021), 125853–125867. https://doi.org/10.1109/ACCESS.2021.3111287
[20] Robert E. Schapire. 2013. Boosting: Foundations and algorithms. *Kybernetes* 42, 1 (2013), 164–166. https://doi.org/10.1108/03684921311295547
[21] Chang Wei Tan, Christoph Bergmeir, François Petitjean, and Geoffrey I. Webb. 2021. Time series extrinsic regression. *Data Mining and Knowledge Discovery* 35, 3 (2021), 1032–1060. https://doi.org/10.1007/s10618-021-00745-9
[22] Keras Team. [n. d.]. Keras Documentation: Keras API reference. https://keras.io/api/
[23] Zhiguang Wang, Weizhong Yan, and Tim Oates. 2017. Time Series classification from scratch with Deep Neural Networks: A strong baseline. *2017 International Joint Conference on Neural Networks (IJCNN)* (2017). https://doi.org/10.1109/ijcnn.2017.7966039
[24] Oliver Watt-Meyer, Noah D. Brenowitz, Spencer K. Clark, Brian Henn, Anna Kwa, Jeremy McGibbon, W. Andre Perkins, and Christopher S. Bretherton. 2021. Correcting weather and climate models by machine learning nudged historical simulations. *Geophysical Research Letters* 48, 15 (2021). https://doi.org/10.1029/2021gl092555
[25] Ayrton Zadra, Keith Williams, Ariane Frassoni, Michel Rixen, Ángel F. Adames, Judith Berner, François Bouyssel, Barbara Casati, Hannah Christensen, Michael B. Ek, and et al. 2018. Systematic errors in weather and climate models: Nature, origins, and Ways Forward. *Bulletin of the American Meteorological Society* 99, 4 (2018). https://doi.org/10.1175/bams-d-17-0287.1

A APPENDIX

A.1

Figure 4 visualizes how convolutional neural network handles a time series of length n and width k. The length is the number of timesteps, and the width is the number of variables in a multivariate time series.

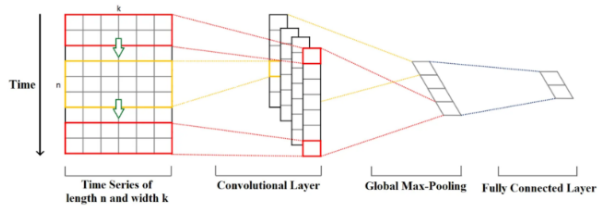


Fig. 4. Sample 1D CNN architecture [13]

Figure 5 shows the network structure for a fully convolutional network developed by Wang et al. [23] for the time classification problem. In 2021 Tan et al. [21] have adjusted the architecture to the regression problems as well where global pooling and Softmax function was replaced with 1 dimensional global average pooling and linear function to predict continuous value.

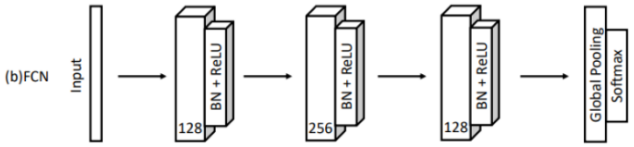


Fig. 5. The structure of the FCN network for the classification problem [23]

B APPENDIX

B.1

Started comparing models

	Model	MAE	MSE	RMSE	\
gbr	Gradient Boosting Regressor	1.025000e-01	0.0249	0.1527	
par	Passive Aggressive Regressor	1.033000e-01	0.0233	0.1496	
lightgbm	Light Gradient Boosting Machine	1.026000e-01	0.0243	0.1517	
omp	Orthogonal Matching Pursuit	1.021000e-01	0.0233	0.1493	
ridge	Ridge Regression	1.021000e-01	0.0233	0.1493	
lr	Linear Regression	1.020000e-01	0.0233	0.1493	
br	Bayesian Ridge	1.020000e-01	0.0233	0.1493	
catboost	CatBoost Regressor	1.032000e-01	0.0246	0.1527	
huber	Huber Regressor	1.000000e-01	0.0231	0.1489	
xgboost	Extreme Gradient Boosting	1.051000e-01	0.0263	0.1565	
ada	AdaBoost Regressor	1.138000e-01	0.0274	0.1595	
rf	Random Forest Regressor	1.131000e-01	0.0292	0.1660	
et	Extra Trees Regressor	1.165000e-01	0.0308	0.1707	
knn	K Neighbors Regressor	1.199000e-01	0.0318	0.1740	
dt	Decision Tree Regressor	1.539000e-01	0.0508	0.2229	
lasso	Lasso Regression	2.223000e-01	0.0877	0.2773	
en	Elastic Net	2.223000e-01	0.0877	0.2773	
llar	Lasso Least Angle Regression	2.223000e-01	0.0877	0.2773	
dummy	Dummy Regressor	2.223000e-01	0.0877	0.2773	
lar	Least Angle Regression	2.093729e+154	inf	inf	

Fig. 6. Results of running PyCaret for univariate time series for one year

B.2

[INFO:root:[15/06/2023T22:42:03.970635Z] Automatically comparing models.

	Model	MAE	MSE	RMSE	\
gbr	Gradient Boosting Regressor	0.1024	0.0255	0.1530	
lr	Linear Regression	0.1008	0.0230	0.1479	
omp	Orthogonal Matching Pursuit	0.1009	0.0230	0.1480	
br	Bayesian Ridge	0.1008	0.0230	0.1479	
ridge	Ridge Regression	0.1008	0.0230	0.1480	
lightgbm	Light Gradient Boosting Machine	0.1046	0.0261	0.1550	
huber	Huber Regressor	0.0990	0.0228	0.1476	
catboost	CatBoost Regressor	0.1075	0.0279	0.1594	
xgboost	Extreme Gradient Boosting	0.1094	0.0294	0.1625	
rf	Random Forest Regressor	0.1133	0.0299	0.1655	
ada	AdaBoost Regressor	0.1230	0.0322	0.1684	
et	Extra Trees Regressor	0.1132	0.0294	0.1648	
par	Passive Aggressive Regressor	0.1201	0.0279	0.1628	
knn	K Neighbors Regressor	0.1340	0.0397	0.1936	
dt	Decision Tree Regressor	0.1486	0.0486	0.2157	
lasso	Lasso Regression	0.2223	0.0877	0.2773	
en	Elastic Net	0.2223	0.0877	0.2773	
llar	Lasso Least Angle Regression	0.2223	0.0877	0.2773	
dummy	Dummy Regressor	0.2223	0.0877	0.2773	
lar	Least Angle Regression	86.3830	84217.7832	187.6737	

Fig. 7. Results of running PyCaret for multivariate time series for one year