Towards Machine Learning Driven Virtual Sensors for Smart Water Infrastructure

Vineeth Maruvada[a], Karamjit Kaur[b], Matt Selway[c] and Markus Stumptner[d]
Industrial AI, University of South Australia, Adelaide, Australia

Keywords: Virtual Sensors, Digital Twins, Water Infrastructure, Artificial Intelligence, Machine Learning, Deep Learning, Long Short Term Memory, XGBoost, Generative Adversarial Networks, Industry 4.0, Water Utilities.

Abstract: Water utilities around the world are under increasing pressure from climate change, urban expansion, and aging infrastructure. To address these challenges, smarter and more sustainable water management solutions are essential. This study explores the use of Machine Learning (ML) to develop Virtual Sensors for smart water infrastructure. Virtual Sensors can complement or replace physical sensors while improving environmental sustainability and enabling reliable and cost-effective Digital Twins (DTs). Our experimental results show that several ML models outperform traditional methods such as Auto-Regressive Integrated Moving Average (ARIMA) in terms of forecast accuracy and timeliness. Among these, Extreme Gradient Boosting (XGBoost) and Long Short-Term Memory (LSTM) offer the best balance between accuracy and robustness. This research provides preliminary evidence that ML models can enable Virtual Sensors capable of delivering short-term forecasts. When successfully implemented, Virtual Sensors can transform water utilities by improving environmental sustainability, operational intelligence, adaptability, and resilience within Digital Twins.

1 INTRODUCTION

Water is essential for public health, economic development, and long-term sustainability of cities. However, modern water utilities face mounting challenges including climate change, rapid population growth, and aging asset infrastructure, resulting in more frequent asset failures, leaks, bursts, and network overflows. This requires an intelligent and proactive approach to water management (Arnell et al., 2019). To address these challenges, utilities need to implement smarter water infrastructure systems that enable early leak detection, accurate demand forecasting, and predictive maintenance to minimize service interruptions.

Physical sensors play an important role as they generate data to efficiently operate water networks. When these data are integrated with Digital Twin (DT), these sensors form the foundation of smart water management, allowing real-time monitoring, simulation of dynamic scenarios, and data-driven decision making (Zekri et al., 2022). However, deploying and maintaining a dense network of physical sensors can be prohibitively expensive and technically challenging, especially when the infrastructure is remote or distributed. Sensor failures and data loss can disrupt network and DT performance, compromising the utility's ability to effectively model and manage water resources.

Artificial Intelligence (AI), and specifically Machine Learning (ML) and Deep Learning (DL), offers a promising solution to mitigate sensor-related disruptions. By learning patterns from historical data, these models can estimate missing or faulty sensor readings, effectively functioning as virtual or soft sensors (Ibrahim et al., 2020). These virtual sensors ensure data continuity when physical sensors fail or provide unreliable readings, thus enhancing the resilience and operational reliability of DT systems. Although previous studies have explored various approaches to develop virtual sensors, there is still a need for systematic evaluation of ML methods for the water industry under real-world operational conditions (Martin et al., 2021). This study aims to fill this gap by benchmarking multiple ML models with statistical methods

[a] https://orcid.org/0000-0001-6644-8834
[b] https://orcid.org/0000-0003-0255-1060
[c] https://orcid.org/0000-0001-6220-6352
[d] https://orcid.org/0000-0002-7125-3289

Maruvada, V., Kaur, K., Selway, M. and Stumptner, M.
Towards Machine Learning Driven Virtual Sensors for Smart Water Infrastructure.
DOI: 10.5220/0013721200003982

In *Proceedings of the 22nd International Conference on Informatics in Control, Automation and Robotics (ICINCO 2025) - Volume 1*, pages 453-460
ISBN: 978-989-758-770-2; ISSN: 2184-2809

to assess their accuracy in replicating physical sensor measurements.

We evaluated the performance of ML models such as XGBoost, LSTM, TadGAN, TSGAN, known for their strong predictive capabilities, and compared them with traditional statistical approaches such as ARIMA. The objective is to assess whether these models can accurately estimate sensor values when physical readings are missing, incomplete, or noisy. Preliminary experiments were conducted using both single-step and multi-step time series forecasting methods. The multi-step results exhibited reduced predictive performance over extended horizons which is consistent recent research highlighting error accumulation in ML-based forecasting models (Marcjasz et al., 2023). Given their limited reliability and scope, multi-step results are not included in this study.

The structure of this paper is as follows: Section 2 reviews Virtual Sensor approaches and research gaps; Section 3 details the Methodology; Section 4 presents Results and Analysis; Section 5 discusses conclusion and future work. To begin, we provide an overview of existing literature to contextualize the development and application of Virtual Sensors and identify key gaps that motivate this study.

2 LITERATURE REVIEW

The following literature review covers the concept of Virtual Sensors, their taxonomy, and concludes by identifying research gaps.

2.1 Virtual Sensors

Virtual Sensors are models that estimate variables that are hard or expensive to measure directly (Kadlec et al., 2009). This approach supports the Industry 4.0 vision of adaptive and resilient systems. As shown in Figure 1, Virtual Sensors combine physical sensor data with ML output to mimic real-world systems within DTs. This enables continuous monitoring and decision making, even when physical sensors fail, helping to build adaptive DTs (Berglund et al., 2023; Zahedi et al., 2024; Shen et al., 2022).

2.2 Taxonomy of Virtual Sensor Models

Based on established practices in industrial systems, environmental monitoring, and recent developments in DTs, virtual sensor implementations fall into the following three broad categories:

- **First-Principles Models:** Based on system physics or chemistry. Includes:

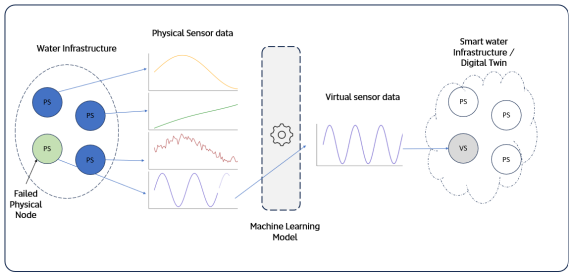

Figure 1: Virtual sensor integration in a water infrastructure digital twin estimating missing or failed sensor values using historical data.

  - *Physics-Based Models* use equations based on physical laws to model sensor behavior.
  - *State Observer-Based Models* that estimate system states using known dynamics
- **Data-Driven Models:** Rely entirely on observed data.
  - *Statistical Techniques:* Linear Regression, Partial Least Squares (PLS), Gaussian Process Regression (GPR)
  - *Stochastic Models:* Kalman filters, Markov models, Bayesian networks
  - *ML/DL Models:* XGBoost, LSTM, CNN, SVM, and GANs
- **Hybrid Models:**
  - *Physics-Informed Neural Networks (PINNs):* Embeds physical laws into neural network training
  - *Other Approaches:* Grey-box models, residual learning, and combinations of physical and ML methods.

This categorization is consistent with several key studies in the field (Martin et al., 2021; Weichert et al., 2019; Psichogios and Ungar, 1992; Karniadakis et al., 2021; González-Herbón et al., 2025), which recognize the spectrum of approaches based on model transparency, data availability, and computational complexity.

Earlier virtual sensor models were based on first-principles methods such as hydraulic equations and state observers, which required domain expertise that often struggled with non-linear or complex systems (Martin et al., 2021; Weichert et al., 2019). To address these limitations, data-driven approaches were developed. Techniques like PLS, Kalman Filters, and GPR showed promise but they do not scale well with large datasets (Rasmussen and Williams, 2006). Recent advances in ML and DL, including Random Forests, SVMs, CNNs, and LSTMs, enable modeling of non-linear patterns and temporal dependencies (Han et al.,

2021; Zhao et al., 2022). Hybrid techniques are a new and evolving area; methods like PINNs combine physical laws with learning algorithms improve both accuracy and interpretability. However, they may introduce additional training and computational complexity (Karniadakis et al., 2021). While these advances have expanded the capabilities of virtual sensors, they also reveal persistent limitations that hinder robust deployment in real-world systems.

2.3 Current Challenges and Research Gaps

Despite advancements in physics-based and statistical modeling methods such as ARIMA, Kalman Filters, and GPR, they often struggle with the dynamic, nonlinear, and nonstationary characteristics of real-world water management. These models typically lack scalability and adaptability, particularly when dealing with high-frequency, multi-sensor data or under conditions of sensor failure (Ibrahim et al., 2020; Berglund et al., 2023). Such limitations hinder the development of reliable DT systems, leading to suboptimal forecasting and inefficient water resource management.

To address these gaps, this study evaluates ML models, specifically XGBoost, LSTM, and GAN variants. By benchmarking them against ARIMA, we assess their ability to provide accurate forecasts. These results align with recent research highlighting the potential of ML in forecasting, anomaly detection, and operational optimization of smart water infrastructure. Furthermore, the integration of virtual sensors within IoT-enabled DT frameworks offers a scalable, adaptive, and cost-effective approach to monitor the water infrastructure in real time (Lakshmikantha et al., 2022), ultimately supporting more resilient and intelligent water management.

3 METHODOLOGY

The following section outlines data collection from physical sensors, the preprocessing steps, and the modeling workflow. It also details the evaluation approach used to assess the performance of various ML techniques.

3.1 Datasets and Preprocessing

This study uses two publicly available real operational datasets collected from flow sensors installed along the Murray River. The first dataset, from sample point 403241, captures significant variations in flow patterns and includes daily measurements from 1 July 2022 to 30 June 2023. The second dataset, from sample point 403242, represents a smaller and more stable system with daily data that span a longer period, from 1 July 2014 to 30 June 2023.

The raw datasets initially contained several attributes, including Sample Point, Description, Latitude, Longitude, SampleDate, Flow Category, Flow, and Comment. To ensure consistency and relevance, the data were preprocessed by removing non-essential fields such as Description, Latitude, Longitude, and Comment. Only records where the Flow Category was set to 'FLOW' were retained. The datasets were then reformatted into a standardized structure with three key fields: SensorID, DateTime, and FlowRate.

By selecting two contrasting datasets, one with dynamic and complex flow behavior and the other with stable operational patterns, this study helps to provide a meaningful comparison of ML techniques under different flow scenarios. We used a univariate forecasting approach that uses historical FlowRate data as input variable.

MinMax Scaling. In univariate time-series forecasting, normalization of the input series is crucial for stable and efficient training. We applied MinMax scaling to the *FlowRate* series to map all historical values to a fixed range, ensuring that the model learns temporal patterns without being influenced by the magnitude of the raw values.

MinMax scaling transforms each observed value y_t in the series $\{y_1, y_2, \ldots, y_T\}$ to a normalized value y'_t within the range $[0,1]$ using the following:

$$y'_t = \frac{y_t - y_{\min}}{y_{\max} - y_{\min}}, \quad \forall t \in \{1, \ldots, T\} \tag{1}$$

where $y_{\min} = \min\{y_1, y_2, \ldots, y_T\}$ and $y_{\max} = \max\{y_1, y_2, \ldots, y_T\}$ represent the minimum and maximum flow values observed during training. This operation is a linear rescaling that preserves temporal dependencies while bounding the values to a consistent scale.

Scaling of the data improves the numerical stability of gradient-based models and helps mitigate issues such as vanishing or exploding gradients. Moreover, it ensures that lagged inputs contribute proportionally during the learning process, allowing the model to focus on sequence patterns rather than absolute magnitudes (Han et al., 2011).

3.2 Feature Engineering

The main feature used for all forecasting models was FlowRate. Although the raw data included additional

attributes such as SensorID and DateTime, the focus was on univariate forecasting, meaning that only past FlowRate values were used as input to predict future values. However, DateTime were used to engineer lagged time steps and sliding windows for training the models. These derived features were not used as inputs and to generate sequences of past FlowRate values to forecast the next. This approach allowed for consistent time alignment and ensured temporal continuity across the datasets.

All models were trained using a one-step-ahead forecasting setup, where the next FlowRate value is predicted based on a fixed window of previous FlowRate readings. This method supports real-time model updates and is suitable for operational scenarios. Although only univariate inputs wsd used for this study, the architecture allows future extension to multivariate forecasting using engineered features like day-of-week, seasonal indicators, or additional sensor readings. Table 1 highlights the key model specific configurations used to reduce prediction errors.

3.3 Model Development

This study implements five forecasting models: ARIMA, LSTM, XGBoost, TadGAN, and TSGAN. The following section describes the architecture, underlying algorithms, and key hyperparameter configurations used for each model.

3.3.1 ARIMA

The AutoRegressive Integrated Moving Average (ARIMA) model is a classical statistical approach for time-series forecasting. It captures linear dependencies in data using a combination of autoregressive terms, differencing operations, and moving averages. ARIMA uses a walk-forward validation strategy, retraining at each time step using newly available observations, making it suitable for real-time forecasting scenarios, although it requires careful parameter tuning for high-frequency or long-horizon tasks. In this study, ARIMA is used as a statistical baseline to evaluate the performance of more recent ML models. Its transparent structure and interpretability provide a valuable reference point for assessing the applicability of data-driven approaches in the context of virtual sensor development (Box et al., 2015).

3.3.2 XGBoost

Extreme Gradient Boosting (XGBoost) is a high-performance ensemble learning algorithm known for its scalability and predictive accuracy in structured datasets. XGBoost operates by using sliding windows of past observations to predict future values for time series forecasting. It constructs additive regression trees through gradient boosting, allowing it to model nonlinear dependencies. Its built-in regularization mechanisms help mitigate overfitting, making it a reliable choice for complex time-series forecasting tasks (Chen and Guestrin, 2016).

Algorithm 1: Unified forecasting procedure for ARIMA, XGBoost, LSTM, TadGAN, and TSGAN.

Data: Preprocessed time-series y
Result: Predicted flow values on the test set
Split y into training and test sets chronologically

ARIMA: begin
Initialize history and parameters (p,d,q)
For each t in test set : fit model, forecast next value $\hat{y}_t$, Store $\hat{y}_t$ in predictions, append y_t to history
end

XGBoost: begin
Generate lag, rolling, and date features
Define hyperparameter search space; Conduct randomized search with time-series cross-validation
Train XGBoost regressor on training data using selected hyper-parameters as per Table 1.
end

LSTM: begin
Create sequences with sliding window T; reshape as $(\text{samples}, T, 1)$
Define, compile and train the model with selected hyper-parameters as per Table 1.
end

TadGAN: begin
Created windowed sequences; Define Generator: LSTM Encoder-Decoder with Repeat Vector
Define Critic: Conv1D + Flatten + Dense
Compile model using selected hyper-parameters as per Table 1.
end

TSGAN: begin
Created windowed sequences; Define Generator: LSTM Encoder-Decoder with Dropout layers
Define Discriminator: Conv1D + Dense + Sigmoid
Compile model using selected hyper-parameters as per Table 1.
end

Train model and predict test values
Inverse scale predictions and test values

Table 1: Key hyperparameter settings used for each forecasting model in this study. The settings were selected through validation or adapted from prior work to ensure fair comparison.

Hyperparameter	ARIMA	XGBoost	LSTM	TSGAN	TadGAN
Order / Estimators / Layers	(2,1,2)	100	LSTM Layers - 2, Units - 50; Dense Layers - 20, Units - 25, 1	LSTM encoder (128, 64)-decoder (64,128)	LSTM encoder (100, 50)-decoder (50,100)
Learning rate	—	0.1	0.001	—	—
Optimizer	—	—	Adam	Adam	Adam
Epochs	—	—	400	400	50
Batch size	—	—	170	170	32
Input time steps (Window size T)	—	—	30	30	30
Loss function	MSE	MSE	MSE	Binary Crossentropy	Binary Crossentropy

3.3.3 LSTM

Long Short-Term Memory (LSTM) networks, a specialized type of recurrent neural network (RNN), are designed to capture long-range temporal dependencies and mitigate the vanishing gradient problem. This makes them particularly effective for modeling sequential patterns in univariate flow rate data. In our implementation, the LSTM model used a fixed input window, referred to as the *timestep*, to predict the next time step's value. This one-step-ahead forecasting framework ensures stable training. Multi-step forecasting would require either recursive predictions or a sequence-to-sequence architecture to extend the output horizon (Hochreiter and Schmidhuber, 1997).

3.3.4 TadGAN

Time-series Anomaly Detection GAN (TadGAN) is designed for unsupervised anomaly detection in time-series data using a GAN architecture with an LSTM-based generator and discriminator. It learns to reconstruct normal patterns in sequential data; anomalies are identified when reconstruction errors exceed a threshold (Geiger et al., 2020). Although, its not a forecasting model, TadGAN can support virtual sensor systems by identifying data inconsistencies or potential faults in real-time monitoring streams.

3.3.5 TSGAN

Time-series GAN (TSGAN) is a generative model tailored for time-series forecasting and synthetic data generation. By learning temporal dependencies, it produces realistic one-step-ahead predictions that can be used to fill in missing values or simulate sensor readings (Smith and Smith, 2020). This makes TSGAN particularly valuable for virtual sensors, where maintaining the continuity and plausibility of the measurements is essential for downstream analytics and decision-making.

3.4 Experimental Setup

All experiments were carried out within the Azure Machine Learning Studio environment using the integrated notebook interface for code execution, data handling, and model evaluation. Each forecasting model was implemented in a dedicated Jupyter notebook and organized by model type (e.g., ARIMA, XGBoost, LSTM, GAN). These notebooks were systematically named to reflect the dataset version and forecasting type, such as single-step or multi-step predictions. Data preprocessing, training, evaluation, and visualization of results were all performed within these notebooks to ensure reproducibility and consistency between models.

Libraries and Model Configuration. A range of libraries supported model development: `statsmodels` was used for ARIMA; `scikit-learn` and `XGBoost` were used for ensemble learning; and `TensorFlow`/`Keras` powered deep learning models including LSTM, TadGAN, and TSGAN. Additional packages such as `pandas`, `NumPy`, `matplotlib`, and `seaborn` facilitated data transformation and visualization. Hyperparameter tuning followed best practices, including walk-forward validation, early stopping, and, where applicable, grid-based optimization to improve generalizability and avoid overfitting.

Data Acquisition. Sensor data was accessed through Azure Machine Learning Studio using a designated Azure datastore. The dataset comprised flow rate measurements tagged with DateTime and Sensor ID. Since each model was designed to operate on data from a single sensor, the dataset was filtered accordingly and sorted in chronological order to maintain temporal integrity essential for time-series forecasting.

Train-Test Splitting. The time-series data was split into training and testing sets using an 80:20 ratio based on temporal order. The first 80% of the data was used to train the models, while the remaining 20% served as the test set. This setup ensured that only past observations were used to forecast future values, mimicking real-world deployment scenarios.

Traditional k-fold cross-validation was not employed, as it violates the temporal sequence and risks data leakage from future to past. Instead, the models were trained once on the training set and evaluated on the unseen test set. Future work may incorporate time-series cross-validation approaches to improve evaluation robustness (Bergmeir et al., 2018).

Evaluation Metrics. Model performance was assessed using three widely accepted error metrics (Hyndman and Koehler, 2006; Makridakis et al., 2018): **RMSE** (Root Mean Squared Error), which penalizes large errors more heavily by squaring them; **MAE** (Mean Absolute Error), which reflects the average magnitude of errors and is robust to outliers; and $\mathbf{R}^2$ (Coefficient of Determination), which indicates the proportion of variance in the observed data explained by the model predictions.

Visual Interpretation. In addition to quantitative evaluation, visual diagnostics were used to interpret model performance, identify error trends, and detect possible non-stationarity or model bias. The forecast results were visualized using `matplotlib` and `seaborn`, displaying training data, actual test observations and predicted values. These plots offered an intuitive assessment of prediction accuracy across models and time horizons. The model execution time for the entire dataset was also recorded to provide context on computational efficiency. This combination of visual and statistical evaluation ensured a comprehensive and transparent evaluation of the effectiveness of the model.

4 RESULTS AND DISCUSSION

In this section, we present and interpret the results for each model output.

4.1 Model Execution Results

Five model outputs ARIMA, XGBoost, LSTM, TadGAN, TSGAN were compared on two real-world datasets.

Table 2: Performance Metrics for Dataset 1 (Sensor 403241).

Model	ARIMA	XGBoost	LSTM	TadGAN	TSGAN
RMSE	2245.5	1647.5	2299.5	4872.3	3068.4
MAE	1226.3	701.8	1341.3	2521.6	1468.5
R^2	0.91	0.94	0.90	0.57	0.83
Exec. Time	20sec	12sec	57sec	43sec	2min 43sec

Tables 2 and 3 present the performance of five models applied to two real-world flow sensor datasets. XGBoost consistently achieved the highest predictive accuracy in both datasets, followed by LSTM and TSGAN. Generative models, particularly TadGAN, showed less stability, especially in Dataset 1, which exhibited more variable flow conditions. This initial analysis highlights the superior performance of the models, particularly XGBoost and LSTM, compared to traditional statistical methods such as ARIMA.

Table 3: Performance Metrics for Dataset 2 (Sensor 403242).

Model	ARIMA	XGBoost	LSTM	TadGAN	TSGAN
RMSE	3925.7	244.2	2525.2	2634.7	2162.6
MAE	1287.9	132.8	940.0	1111.3	884.6
R^2	0.86	0.99	0.91	0.90	0.93
Exec. Time	3min 9sec	14sec	4min 5sec	3min 59sec	15min 23sec

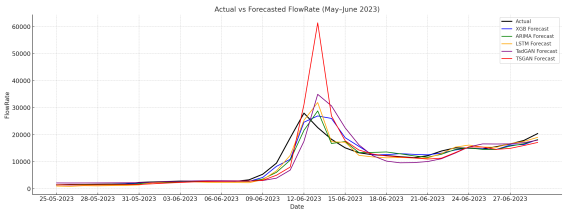

Figure 2: Results Visualization for Dataset 1.

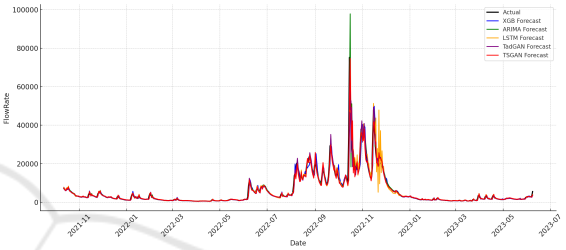

Figure 3: Results Visualization for Dataset 2.

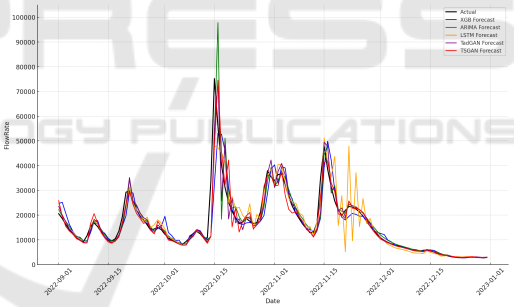

Figure 4: Results Visualization for Dataset 2 (Zoomed: Sep 2021–Jan 2022).

For Sensor 403241 (Dataset 1) and Sensor 403242 (Dataset 2), XGBoost achieved the best results, with RMSE values of 1647.5 and 244.2, and R^2 scores of 0.94 and 0.99, respectively. LSTM and TSGAN also performed well, particularly in Dataset 2, where TSGAN demonstrated its ability to model complex, nonlinear patterns with an RMSE of 2162.6 and an R^2 of 0.93. In contrast, ARIMA showed consistently higher

error rates and lower explanatory power, reflecting its limitations in capturing dynamic water usage behaviors. These findings support the growing emphasis in the literature on the role of ML in enhancing forecasting accuracy for infrastructure systems. In particular, the model execution time for XGBoost was significantly lower compared to other models, making it a strong candidate for real-time deployment.

Figure 2 illustrates the forecast output for Dataset 1 during the May–June 2023 period. ARIMA successfully captured the general trend, but lagged slightly during the peak flow. XGBoost tracked the actual data, especially during both the rising and falling phases of the event. LSTM produced smooth forecasts and maintained good alignment throughout, though it slightly overestimated the peak. TadGAN followed the flow trend relatively well but overpredicted the peak value. TSGAN exhibited the largest overestimation at the peak, reducing its overall accuracy in this period. Overall, XGBoost and LSTM offered the most reliable and consistent predictions in this scenario.

Similar patterns were observed in Dataset 2, as shown in Figures 3 and 4. ARIMA captured the general flow trend, but struggled to respond accurately during peak events, especially with sharp surges. XGBoost consistently aligned with the actual flow, and performed well across both stable and volatile periods. LSTM produced smooth forecasts overall but exhibited increased noise and fluctuations in certain intervals, due to overfitting on local noise when the training data contains high variance, combined with its strong temporal memory. TadGAN followed the trend reasonably but introduced fluctuations around high-flow events. TSGAN demonstrated strong alignment with the actual values, particularly during peak periods, and outperformed other generative models. Overall, XGBoost and TSGAN provided the most reliable forecasts for Dataset 2.

4.2 Model Performance Insights

Our evaluation of models reveals that modern ML approaches, particularly XGBoost and LSTM, consistently outperformed traditional statistical methods such as ARIMA in flow prediction tasks. XGBoost achieved the highest accuracy across both datasets, with an RMSE of 1647.50 and R^2 of 0.94 in Dataset 1, and an even stronger RMSE of 244.17 and R^2 of 0.99 in Dataset 2. LSTM also performed robustly, effectively capturing temporal patterns. Among generative models, TSGAN showed potential in modeling nonlinear dynamics, achieving an R^2 of 0.93 on Dataset 2 but struggled with irregular patterns in Dataset 1. ARIMA, by contrast, produced consistently higher errors and lower R^2 scores, highlighting its limitations in dynamic systems.

While XGBoost and LSTM were generally reliable, their effectiveness varied by dataset, underscoring the importance of tailoring virtual sensor models to specific environments. Regular retraining based on evolving demand is critical to ensure sustained accuracy. Integrating such models within DT frameworks would enable automated updates and support real-time operational monitoring.

These results suggest that ML models can serve as effective tools for immediate data imputation and near term prediction. However, more work is needed to extend these benefits to long-term prediction scenarios. Overall, these insights offer practical guidance for water utilities seeking to adopt ML and DL techniques. By prioritizing robust models like XGBoost and LSTM, and exploring the integration of generative approaches for anomaly detection and data gap filling, utilities can significantly enhance the resilience, scalability, and decision making capabilities of smart water infrastructure.

5 CONCLUSION AND FUTURE WORK

This study investigates the potential of machine learning-based virtual sensors to replicate physical sensor output using historical flow data from two key locations. Preliminary results show that short-term flow forecasts can be generated accurately and efficiently using models such as XGBoost and LSTM. Both models outperformed traditional statistical methods such as ARIMA in terms of predictive accuracy and computational efficiency. In particular, XGBoost delivered the best balance between speed and performance, positioning it as a practical solution for filling data gaps and supporting short-term forecasts.

Although these results are based on historical data under known conditions, they demonstrate the strong potential of XGBoost and LSTM for developing virtual sensors in water infrastructure. Future work should focus on validating these models using real-time water network data, integrating insights from correlated sensors, and enhancing multi-step forecast capabilities. With further refinement, ML-based virtual sensors could play a critical role in improving the resilience, and responsiveness of smart water infrastructure.

REFERENCES

Arnell, N. W. et al. (2019). Global and regional impacts of climate change at different levels of global temperature increase. *Global Environmental Change*, 58:101–113.

Berglund, E. Z., Shafiee, M. E., Xing, L., and Wen, J. (2023). Digital twins for water distribution systems. *Journal of Water Resources Planning and Management*, 149(3):02523001.

Bergmeir, C., Hyndman, R. J., and Koo, B. (2018). A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Computational Statistics & Data Analysis*, 120:70–83.

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015). *Time Series Analysis: Forecasting and Control*. Wiley, Hoboken, NJ, 5th edition.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM.

Geiger, A., Liu, D., Alnegheimish, S., Cuesta-Infante, A., and Veeramachaneni, K. (2020). Tadgan: Time series anomaly detection using generative adversarial networks. In *IEEE Int'l. Conf. on Big Data (Big Data '20')*, pages 33–43. IEEE.

González-Herbón, R., González-Mateos, G., Rodríguez-Ossorio, J., Prada, M., Morán, A., Alonso, S., Fuertes, J., and Domínguez, M. (2025). Assessment and deployment of a lstm-based virtual sensor in an industrial process control loop. *Neural Computing and Applications*, 37(17):10507–10519.

Han, J., Kamber, M., and Pei, J. (2011). *Data Mining: Concepts and Techniques*. Elsevier, Burlington, MA, 3rd edition.

Han, Z., Zhao, J., Leung, H., Ma, K.-F., and Wang, W. (2021). A review of deep learning models for time series prediction. *IEEE Sensors Journal*, 21(6):7833–7848.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hyndman, R. J. and Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688.

Ibrahim, T., Omar, Y., and Maghraby, F. A. (2020). Water demand forecasting using machine learning and time series algorithms. In *2020 International Conference on Emerging Smart Computing and Informatics (ESCI)*, pages 325–329. IEEE.

Kadlec, P., Gabrys, B., and Strandt, S. (2009). Data-driven soft sensors in the process industry. *Computers & Chemical Engineering*, 33(4):795–814.

Karniadakis, G. E. et al. (2021). Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440.

Lakshmikantha, V., Hiriyannagowda, A., Manjunath, A., Patted, A., Basavaiah, J., and Anthony, A. A. (2022). Iot based smart water quality monitoring system. *Materials Today: Proceedings*, 51:1283–1287.

Makridakis, S., Spiliotis, E., and Assimakopoulos, V. (2018). Statistical and machine learning forecasting methods: Concerns and ways forward. *PLoS ONE*, 13(3):e0194889.

Marcjasz, G., Narajewski, M., Weron, R., and Ziel, F. (2023). Distributional neural networks for electricity price forecasting. *Energy Economics*, 125:106843.

Martin, D., Kühl, N., and Satzger, G. (2021). Virtual sensors. *Business & Information Systems Engineering*, 63:315–323.

Psichogios, D. and Ungar, L. (1992). A hybrid neural network-first principles approach to process modeling. *AIChE Journal*, 38(10):1499–1511.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press.

Shen, Y. et al. (2022). Digital twins for smart water management: Framework and case studies. *Water Research*, 218:118449.

Smith, K. E. and Smith, A. O. (2020). Conditional gan for timeseries generation. *arXiv preprint arXiv:2006.16477*.

Weichert, D. et al. (2019). A review of machine learning for the optimization of production processes. *International Journal of Advanced Manufacturing Technology*, 104(9–12):3663–3682.

Zahedi, F., Alavi, H., Majrouhi Sardroud, J., and Dang, H. (2024). Digital twins in the sustainable construction industry. *Buildings*, 14(11):3613.

Zekri, S., Jabeur, N., and Gharrad, H. (2022). Smart water management using intelligent digital twins. *Computing and Informatics*, 41(1):135–153.

Zhao, Z., Chen, W., Wu, X., Chen, P., Liu, J., Chen, J., and Deng, S. (2022). Deep learning for time series forecasting: A survey. *Artificial Intelligence Review*, 55:4099–4139.