# Multivariate Timeseries Forecasting – Based on Deep Neural Networks Models in Industrial Control Systems

Amr Alfayoumy
*Electronic and Computer Engineering Department*
*Nile University*
*Giza, Egypt*
a.alfayoumy@nu.edu.eg

***Abstract*— Recent advancements in deep learning have shown promising results in forecasting and anomaly detection for industrial control systems (ICS), particularly in the context of secure water treatment plants (SWaT). Accurate forecasting of system variables such as flow rates, pressures, and temperatures is critical for ensuring operational efficiency, safety, and sustainability in Industrial Control Systems (ICS) such as the Secure Water Treatment (SWaT) testbed. The complex relationship of interconnected variables and dynamic environmental conditions pose unique challenges that traditional forecasting methods often fail to address. This research aims to investigate and compare the effectiveness of three deep learning architectures – Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), and Bidirectional Gated Recurrent Unit (BiGRU) – in forecasting sensor and actuator readings for the SWaT testbed for early detection of anomalies caused by malfunctions or cyberattacks. Two ensemble models are also proposed, combining the predictions of the individual models to potentially improve overall forecasting performance. The results demonstrate the superior performance of the BiGRU model, which achieved the lowest MAE of 1.1968, MSE of 44.8929, and RMSE of 6.7002. The BiGRU model also exhibited the highest R-squared coefficient of 0.9990 and the highest percentage of values within a 15% tolerance range at 96.11%. Quantile mapping was effectively applied as a post-processing technique to further improve the accuracy of select features across the different models. The ensemble model designed to minimize the MAE achieved the lowest MAE (1.1090), while the ensemble model designed to maximize the PWT achieved the highest PWT (96.80%). This research addresses a critical need in the domain of ICS by leveraging advanced deep-learning techniques to improve forecast accuracy and empower decision-makers with timely, actionable insights for effective system management and operation.**

***Keywords—industrial control system, secure water treatment, forecasting, multivariate time series, deep learning, neural networks, encoder-decoder models, LSTM, BiLSTM, BiGRU, ensemble models, recurrent neural networks.***

## I. INTRODUCTION

In recent years, the integration of advanced computational techniques, particularly deep learning models, has revolutionized the field of predictive analytics in various industrial control systems (ICS) [1], [2], [3], [4], [5], [6]. One such critical area is the management and operation of secure water treatment (SWaT) systems, where accurate forecasting of system variables is paramount for ensuring operational efficiency, safety, and sustainability. The ability to predict fluctuations in key parameters such as flow rates, pressures, and temperatures enables proactive decision-making, preventive maintenance, and rapid response to potential anomalies caused by malfunctions, cyberattacks, or other disruptions [7], [8], [9], [10].

However, forecasting in SWaT systems poses unique challenges due to the complex relationship of several interconnected variables, dynamic environmental conditions, and the need for real-time adaptability. Traditional forecasting methods often fall short in capturing the intricate temporal dependencies and nonlinear relationships inherent in SWaT data, leading to suboptimal performance and limited predictive accuracy. In this context, the application of deep learning models offers a promising avenue for improving forecast accuracy and enhancing system resilience.

This research aims to explore and contrast the ability of three deep learning architectures – Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), and Bidirectional Gated Recurrent Unit (BiGRU) – in forecasting SWaT readings. These models are chosen for their ability to capture temporal dynamics and handle multivariate time series data, making them well-suited for the complex forecasting tasks inherent in SWaT systems.

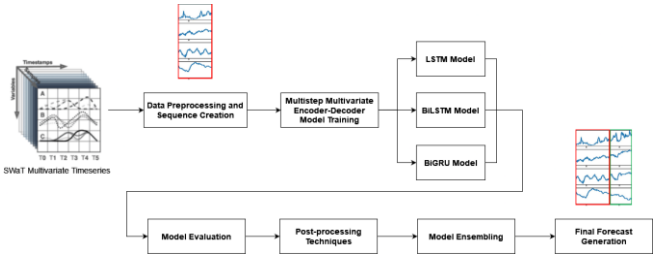



Fig. 1. Overview of the SWaT Forecasting Framework

The research methodology involves a comprehensive analysis of each model's architecture, training process, and

evaluation metrics, including Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R-squared coefficient, and the percentage of values within specified tolerance limits. Additionally, post-processing techniques such as quantile mapping are employed to further enhance forecast accuracy by aligning model predictions with the distribution of original data.

Through rigorous experimentation and comparative analysis, this research seeks to identify the model(s) that demonstrate superior performance in forecasting SWaT readings with high accuracy and reliability. Furthermore, insights gained from this study can inform the development of robust predictive analytics solutions tailored to the unique requirements of SWaT systems, ultimately contributing to enhanced operational efficiency, resource optimization, and resilience in water treatment infrastructure.

In summary, this research addresses a critical need in the domain of secure water treatment by leveraging advanced deep-learning techniques to improve forecast accuracy and empower decision-makers with timely, actionable insights for effective system management and operation.

## II. Literature Review

Emerging deep learning technologies have demonstrated promising outcomes in forecasting and identifying anomalies within ICS, especially in the context of SWaT facilities. This literature review examines current approaches and challenges in applying deep learning models to multivariate time series data from ICS sensors and actuators.

Tian et al. [11] propose STADN (Spatial and Temporal Anomaly Detection Network), which leverages both spatial and temporal dependencies in multivariate time series data. Their approach combines graph attention networks to capture spatial relationships between variables and long short-term memory (LSTM) networks to model temporal dependencies. STADN predicts future sensor behavior by incorporating historical data from the sensor and its neighbors, detecting anomalies based on prediction errors. The authors report state-of-the-art performance on real-world datasets, addressing the challenge of limited labeled anomaly data in industrial systems.

Challú's [12] work focuses on developing efficient and robust deep learning architectures for time series forecasting and anomaly detection. The research explores dynamic latent space representations to improve model performance and robustness. Additionally, Challú introduces a novel scalable and interpretable model for multi-step forecasting based on non-linear frequency decomposition, with applications in energy and healthcare domains. This work addresses key challenges in adopting state-of-the-art methods, including handling distribution shifts, missing data, computational complexity, and interpretability.

Park et al. [13] propose two approaches for anomaly detection in the SWaT dataset: Functional Data Analysis (FDA) and Autoencoder-based methods. These techniques aim to capture underlying forecasting error patterns generated by Mixture Density Networks (MDNs). The authors demonstrate that their methods outperform baseline approaches such as cumulative sum (CUSUM) and static thresholding, highlighting the potential of advanced statistical and deep learning techniques in ICS anomaly detection.

Hu and Liu [14] present an attack prediction and recognition method specifically designed for water treatment systems. Their approach combines a one-dimensional convolutional neural network (1D-CNN) for time series prediction with CUSUM statistics for anomaly detection. Experimental results on the SWaT Test Bench show that 1D-CNN exhibits superior predictive performance compared to other models, while CUSUM successfully detects and identifies all listed attacks based on statistical bias.

These studies collectively demonstrate the potential of deep learning models in forecasting and anomaly detection for ICS, particularly in water treatment systems. Common themes include:

- Leveraging spatial and temporal dependencies in multivariate time series data
- Addressing the challenge of limited labeled anomaly data
- Developing scalable and interpretable models
- Combining deep learning techniques with statistical methods for improved performance

While numerous studies have explored the application of machine learning techniques in industrial control systems, there remains a significant gap in the comparative analysis of state-of-the-art deep learning architectures for forecasting sensor and actuator readings in Secure Water Treatment (SWaT) testbeds. Previous research has primarily focused on individual model performance or traditional forecasting methods, which often fall short in addressing the complex, interconnected nature of SWaT systems. Moreover, the literature lacks a comprehensive evaluation of models such as Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), and Bidirectional Gated Recurrent Unit (BiGRU) in this specific context. Additionally, there is limited exploration of post-processing techniques like quantile mapping and ensemble modeling to further enhance forecast accuracy in SWaT environments. This study aims to fill these gaps by providing a thorough comparison of these deep learning architectures, incorporating post-processing methods, and proposing novel ensemble models. By doing so, this research seeks to contribute valuable insights into the most effective approaches for accurate forecasting in SWaT systems, ultimately improving operational efficiency, safety, and sustainability in industrial control systems.

## III. Methodology

### A. Dataset

The Secure Water Treatment (SWaT) testbed was developed by researchers and students at the Singapore University of Technology and Design [15] as part of their research on Cyber-Physical systems security. It is an operational small-scale water treatment plant with six process stages capable of producing 5 gallons per minute of double-filtered water. The system architecture is a distributed control network where each stage is managed by programmable logic controllers. The SWaT A1 & A2-Dec 2015 dataset, utilized in this study, encompasses 11 days of continuous operation, with the first 7 days under normal conditions and the remaining 4 days including 36 simulated attacks. The dataset characteristics include: 11 days of continuous operation (7 under normal conditions and 4 with attack scenarios with data records labeled as normal or abnormal). The dataset also collected network traffic and data from all 51 sensors and

actuators which consider the intent space of a Cyber-Physical System.

### B. Data Preprocessing

In this section, we utilize Python to read and preprocess the multiple Excel files containing the SWaT 2015 dataset. The data, which encompasses both normal operation and attack instances within the SWaT system, is consolidated into a single DataFrame for easier manipulation and analysis.

The dataset provides detailed information about the SWaT system, including 946,719 timestamped observations of sensor readings and control signals, spanning the period from 2015-12-22 16:00:00 to 2016-01-02 14:59:59. The data is stored in a DataFrame with 53 columns, comprising 51 features in addition to the Timestamp and Normal/Attack columns, with 25 float64 columns and 26 int64 columns. The entire dataset occupies approximately 390 MB of memory.

An analysis of the unique values within the dataset provides insights into its diversity. Columns with few unique values likely represent categorical or binary features, while those with many unique values indicate continuous or granular measurements. Certain sensors exhibit a wide range of unique values, reflecting variability in measurements, while binary columns have a limited number of unique values, reflecting discrete operational states. Out of the 51 features in the dataset, 6 were dropped due to having only one unique value: P202, P401, P404, P502, P601, P603.

The data is reshaped to match the input requirements of the LSTM model. It's then normalized using MinMaxScaler. The MinMaxScaler is a feature scaling technique provided by scikit-learn that scales and translates each feature individually such that it is in the specified range, typically between 0 and 1.

MinMaxScaler is a useful technique for scaling features that do not follow a Gaussian distribution or have varying scales. It computes the minimum and maximum values of each feature in the training dataset and scales them to a specified range, preserving the relationships between data points. This method is robust to outliers and ensures that scaled features remain within predefined ranges, aiding interpretation and decision-making in SWaT forecasting applications.

MinMaxScaler is a useful scaling technique for SWaT forecasting applications, as it preserves the interpretability of the data and ensures all features are normalized to a common scale. This is particularly important when dealing with non-Gaussian data or features with predefined ranges. MinMaxScaler's consistency, interpretability, and normalization benefits make it a preferred choice for SWaT forecasting, where the interpretability of sensor readings is crucial for system monitoring and control.

### C. Model Preparation

#### 1) Sequence Generation

The function create_sequences(data, input_seq_length, output_seq_length) is defined to create input-output sequences for the forecasting models. The sequencing technique used here involves using a time series forecasting model where sequences of input_seq_length seconds are taken with a specified step size to predict the next output_seq_length seconds. The step size determines the overlapping between subsequent sequences [16].

After experimenting with various combinations of sequence length values, the following parameters were determined to be optimal for the SWaT forecasting model:

- Input sequence length: 5 minutes (300 seconds) of input sequence length.
- Output sequence length: 2 minutes (120 seconds) of output sequence length.
- Step size: 1 second (training), 120 seconds (inference).

Using a step size of 1 during training generates the maximum number of training sequences from the dataset, which is beneficial for model learning. The model can learn from the finer granularity of the data, capturing more detailed temporal dynamics. Additionally, using a step size of 120 during inference reduces the number of predictions we need to make, which speeds up the process. This approach prevents unnecessary overlap in predictions, making the inference process more straightforward and less computationally intensive. By training with a step size of 1, we leverage the maximum amount of data to train the model more effectively. During inference, using a step size of 120 balances the need for efficiency with the quality of predictions, ensuring that the model remains responsive and performant.

These values were determined through trial and error, where different combinations were tested to achieve the best performance and convergence of the model. By iteratively adjusting these parameters and evaluating the model's performance, the optimal values were identified to effectively capture the temporal dynamics of the data and facilitate efficient training.

#### 2) Batch Generation

Another function batch_generator(data, input_seq_length, output_seq_length, batch_size) is defined to generate batches of training data for the model. The batch generator function plays a crucial role in managing memory usage during model training, particularly when dealing with large datasets where it helps avoid flooding the RAM.

Instead of loading the entire dataset into memory at once, the batch generator function processes data in smaller chunks or batches. This means that only a subset of the data, equal to the batch size, needs to be loaded into memory at any given time. As a result, it efficiently utilizes available memory resources without overwhelming them.

The batch generator function is a key component in the training process. It generates batches of data on-the-fly, reads and processes the data, and passes it to the model for training. This iterative approach prevents memory overflow by discarding each batch after processing and loading the next one. Batch generator functions are scalable and can adapt to datasets of varying sizes by adjusting the batch size. By feeding data sequentially, the batch generator function ensures continuity in the training process, allowing the model to learn gradually over multiple epochs without interruptions due to memory constraints.

#### 3) Callbacks

Callbacks including ModelCheckpoint, EarlyStopping, and ReduceLROnPlateau are set up to monitor the model during training for saving the best weights, early stopping if the validation loss doesn't improve, and reducing the learning rate if the loss plateaus.

*a) ModelCheckpoint*

The ModelCheckpoint callback is used to save the model's weights during training, particularly the weights that result in the best performance on the validation dataset [17].

Parameters:

- filepath: Specifies the path where the model weights will be saved.
- save_weights_only: If set to False, the entire model (including architecture and weights) will be saved. If set to True, only the weights will be saved. This parameter is set to False.
- monitor: Specifies the metric to monitor for determining the best weights. In this case, 'val_loss' indicates that the validation loss will be monitored.
- mode: Defines whether to minimize or maximize the monitored metric. Here, 'min' indicates that the goal is to minimize the validation loss.
- save_best_only: This is set to True, which means only the weights corresponding to the best performance on the monitored metric will be saved.

*b) EarlyStopping*

The EarlyStopping callback is used to halt the training process if the monitored metric (validation loss) stops improving after a certain number of epochs (patience) [18].

Parameters:

- monitor: Specifies the metric to monitor for early stopping. In this case, 'val_loss' indicates that the validation loss will be monitored.
- patience: Specifies the number of epochs to wait before stopping training if there is no improvement in the monitored metric. This parameter is set to 5 epochs.
- mode: Defines whether to minimize or maximize the monitored metric. Here, 'min' indicates that the goal is to minimize the validation loss.

*c) ReduceLROnPlateau*

The ReduceLROnPlateau callback is used to dynamically adjust the learning rate during training if the monitored metric (validation loss) plateaus, i.e., stops improving [19].

Parameters:

- monitor: Specifies the metric to monitor for learning rate adjustment. In this case, 'val_loss' indicates that the validation loss will be monitored.
- factor: Specifies the factor by which the learning rate will be reduced. In our case, a factor of 0.1 means reducing the learning rate by 10%.
- mode: Defines whether to minimize or maximize the monitored metric. Here, 'min' indicates that the goal is to minimize the validation loss.
- patience: Specifies the number of epochs to wait before reducing the learning rate if there is no improvement in the monitored metric. This is set to 2 epochs.
- min_lr: Specifies the minimum learning rate that the callback can reduce to, which is set to 1e-4. This prevents the learning rate from becoming too small.

During training, the ModelCheckpoint callback saves the weights of the model whenever there is an improvement in the validation loss. The EarlyStopping callback monitors the validation loss and stops training if there is no improvement after a certain number of epochs (patience). The ReduceLROnPlateau callback dynamically adjusts the learning rate if the validation loss plateaus, helping the model converge faster.

*4) Evaluation Metrics*

Mean Absolute Error (MAE) measures the average of the absolute differences between predicted and actual values. It is less sensitive to outliers compared to Mean Squared Error (MSE) since it does not square the errors. MAE provides a straightforward measure of average prediction error and is often preferred when outliers are present [20].

Mean Squared Error (MSE) measures the average of the squared differences between predicted and actual values. MSE penalizes larger errors more heavily due to the squaring operation, making it sensitive to outliers [21]. It is commonly used in regression tasks to quantify the average squared deviation between predicted and true values.

Root Mean Squared Error (RMSE) measures the square root of the average of the squared differences between predicted and actual values. RMSE is similar to MSE but provides an interpretable scale in the same units as the target variable, offering a measure of prediction accuracy while maintaining the interpretability of the original scale [20].

R-squared is a statistical measure that quantifies the proportion of variance in the dependent variable that is predictable from the independent variable(s). It ranges from 0 to 1, with higher values indicating a better fit of the model to the data. R-squared provides an indication of how well the model explains the variability of the target variable and is often used for model comparison [22].

The percentage of values within tolerance (PWT) evaluation metric assesses the performance of a multi-regression model by calculating the percentage of its predictions that fall within a specified tolerance range of the actual values. It involves determining the number of predictions that meet the tolerance criteria, dividing by the total number of predictions, and reporting the result. This provides insight into the model's accuracy within a certain margin of error, which can be useful for evaluating its suitability for practical applications that allow for a degree of tolerance.

*5) Loss Function*

The Mean Absolute Error (MAE) loss function, also known as L1 loss, is a widely used metric in regression tasks to measure the average absolute difference between predicted and actual values. It is less sensitive to outliers compared to Mean Squared Error, making it suitable for datasets with outliers. MAE directly represents the average absolute deviation between predictions and true values, facilitating easy interpretation in the problem domain.

MAE loss is a useful metric for evaluating regression models, as it measures the average absolute deviation between predicted and true values [23]. It is differentiable and allows for efficient optimization, making it well-suited for model training. In the context of the SWaT forecasting models, minimizing the MAE loss helps optimize the model parameters to capture the underlying patterns in the data and improve prediction accuracy.

*6) Optimizer*

The Adam optimizer [24] stands out for its adaptive learning rates and self-tuning ability, which allow it to converge faster and handle varying parameter scales effectively. Adam also incorporates momentum, similar to SGD with momentum, which further accelerates convergence by helping the optimizer maintain direction even with sparse or noisy gradients. The combination of adaptive learning rates and momentum provides improved convergence speed and reduced oscillations during training.

Additionally, Adam benefits from the concepts of RMSprop, utilizing moving averages of both gradients and squared gradients to scale the learning rates differently for each parameter. This adaptive learning rate method provides stability and efficiency by adjusting learning rates based on the magnitudes of gradients.

One of the notable features of Adam is its self-tuning ability. Throughout the training process, Adam automatically adjusts hyperparameters such as learning rate, momentum, and decay rates based on observed gradients [24]. This adaptability makes Adam suitable for a wide range of deep learning tasks and architectures, reducing the need for manual hyperparameter tuning and improving convergence speed and performance.

*7) Activation Functions*

In GRU (Gated Recurrent Unit) and LSTM (Long Short-Term Memory) networks, the use of tanh as the activation function and sigmoid as the recurrent activation function is rooted in their complementary properties that help stabilize and improve the learning process.

The tanh activation [25] function maps input values to a zero-centered range, which facilitates faster learning and effective gradient flow in neural networks, particularly for recurrent neural networks where the same parameters are used across time steps.

The sigmoid function [25] is used in recurrent neural network gates to control the flow of input information and previous states. Its output can be interpreted as a probability or gate, allowing selective passage of information. Although the sigmoid function has drawbacks like gradient saturation, its use in gates is advantageous because it ensures a smooth and differentiable gating mechanism.

The tanh activation function helps in updating the cell state by ensuring that the values are in the range [-1, 1], which helps in keeping the gradients within a manageable range and contributes to the network's ability to retain information over long sequences. Whereas, the sigmoid function in gates modulates how much of the previous state and new information to keep, update, or forget. By mapping the inputs to the range [0, 1], it effectively controls the flow of information in a smooth, differentiable manner. In summary, the combination of tanh and sigmoid leverages the strengths of both functions: tanh for managing state values and sigmoid for gating mechanisms, contributing to the overall stability and effectiveness of GRU and LSTM networks in sequence modeling tasks.

*8) Quantile Mapping*

Quantile mapping is a statistical technique used to adjust the distribution of forecasted or modeled data to match the distribution of observed data. It involves mapping the cumulative distribution function (CDF) of the forecasted data to the CDF of the observed data [26]. This adjustment helps to correct biases or discrepancies between the forecasted and observed data, particularly in cases where the forecasted data exhibit systematic errors or do not adequately capture the variability present in the observed data.

In the context of time series forecasting, such as in the SWaT system readings, quantile mapping can be applied to improve the accuracy of forecasted sequences by aligning them with the distribution of the actual data. By adjusting the forecasted values based on the distributional characteristics of the observed data, quantile mapping aims to enhance the fidelity of the forecasts and improve their suitability for operational decision-making or analysis.

*D. Forecasting Models*

*1) Baseline Model*

A baseline model serves as a reference point for evaluating the performance of more complex models. It's usually a simple model that doesn't involve sophisticated techniques or algorithms. In forecasting tasks, a common baseline is to predict the mean or median of the target variable for all future time points.

In this context, the baseline model involves predicting the mean of each feature column across the training data and repeating these mean values for the length of the test data. This simple approach assumes that future observations will follow the same average pattern as observed in the training data.

The code calculates the mean of each column in the training data using np.mean(train_data.values, axis=0). These mean values are then repeated for the length of the test data using baseline_predictions = np.full_like(test_data.values, fill_value=column_means).

Afterward, the performance of the baseline model is evaluated using various metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared. These metrics provide insights into how well the baseline model performs compared to the actual test data.

*2) Multistep Multivariate Encoder-Decoder LSTM Model*

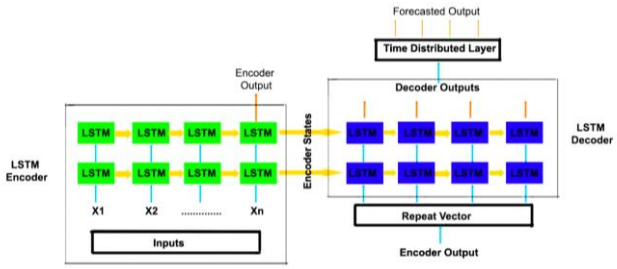



Fig. 2. Encoder-Decoder LSTM Architecture [27]

This section introduces the enhancement of the “vanilla” LSTM model into a multistep multivariate encoder-decoder architecture for SWaT readings forecasting. Unlike the traditional LSTM, this model consists of an encoder and a decoder [28]. LSTM was introduced in 1997 to mitigate the vanishing gradient problem in recurrent networks [29]. The encoder processes and encodes the input sequence, while the decoder utilizes LSTM units to predict each element of the output sequence based on the encoded input. Although both approaches yield sequence outputs, the use of an LSTM in the decoder allows for enhanced prediction accuracy by

considering previous predictions and maintaining the internal state during sequence generation. The updated model incorporates each of the time series variables to predict future readings of the SWaT sensors and actuators.

By presenting each one-dimensional time series as a distinct input sequence to the model, the LSTM generates an internal representation of each input, which is then interpreted collectively by the decoder. This multivariate approach proves advantageous for forecasting tasks where the output sequence depends on observations from multiple features across preceding time steps.

*a) Model Architecture*

The LSTM model architecture is defined. It consists of one LSTM layer with 400 units in the Encoder step, followed by a RepeatVector layer, then one LSTM layer with 200 units in the Decoder step, and finally a TimeDistributed layer with Dense output. The RepeatVector Layer replicates the encoded representation of the input sequence to align with the desired output sequence length. It ensures the encoded representation is compatible with the decoder input. As for the TimeDistributed Dense Layer, it applies a dense (fully connected) layer to each time step of the decoder output sequence. It produces the final output sequence with the number of features as output units. The model is compiled with Adam optimizer and Mean Absolute Error (MAE) loss function.

The LSTM (Long Short-Term Memory) model architecture was meticulously designed to capture the temporal dependencies within the SWaT (Secure Water Treatment) system data. Comprising an encoder-decoder framework, the model incorporates two LSTM layers for sequence processing. The encoder, equipped with 400 units, encodes the input sequence, while the decoder, featuring 200 units, predicts future readings based on the encoded input. This architecture enables the model to retain internal state information and consider previous predictions during sequence generation, contributing to its enhanced forecasting accuracy.

The output layer, implemented as a TimeDistributed dense layer with 45 units, ensures compatibility with the multivariate nature of the SWaT dataset, which consists of 51 features, 6 of which were dropped for having only one unique value. With a total of 1,203,445 trainable parameters (4.59 MB), the model possesses the capacity to capture intricate patterns and relationships within the data, facilitating comprehensive forecasting capabilities.

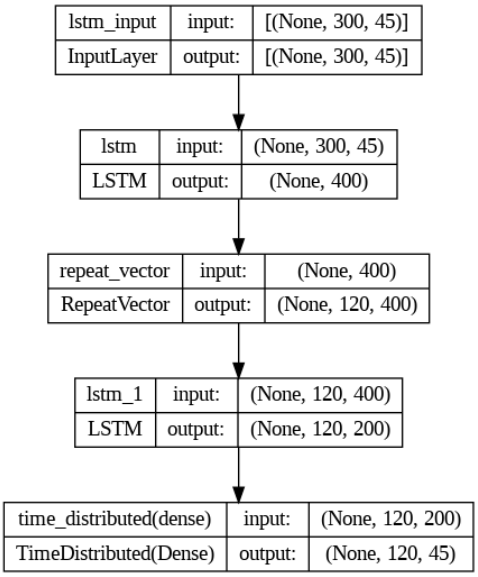



Fig. 3. LSTM Model Layers

*b) Model Hyperparameters*

The model hyperparameters are as follows: The model consists of two LSTM layers, with the first layer having 400 memory cells and the second layer having 200 memory cells. The batch size is set to 64, and there is no dropout or recurrent dropout applied to the LSTM layers. The activation function used for both LSTM layers is tanh, and the recurrent activation function for both LSTM layers is sigmoid. The input sequence length is 300 seconds, and the kernel initializer, recurrent initializer, and bias initializer are set to GlorotUniform, Orthogonal, and Zeros, respectively. The unit forget bias is enabled, and the LSTM layers' states are not preserved between batches. The first LSTM layer returns only the last output, while the second LSTM layer returns the full sequence of outputs. The input sequence is not processed backwards, and the LSTM loop is not unrolled. The model is trained for 100 epochs but stops once the EarlyStopping callback is triggered. The input layer shape is specified by the batch_input_shape parameter as (None, 300, 45), and a RepeatVector layer is used to repeat the input 120 times for an output sequence of 120 seconds. Finally, a TimeDistributed layer is applied, which contains a Dense layer with 45 units for 45 features with a linear activation function. The Adam optimizer with default settings is utilized, while Mean Absolute Error (MAE) serves as the loss function for evaluating the deviation between predicted and actual values. Moreover, the learning rate undergoes dynamic adjustments during training via the ReduceLROnPlateau callback, reducing by a factor of 0.5 if the validation loss stagnates for 2 consecutive epochs.

*c) Model Training*

The model underwent training for 68 epochs, triggering the Early Stopping callback, and utilizing sequences derived from both the training and testing datasets for model refinement. This iterative process allowed the model to adjust its parameters and improve its performance over time. Throughout the training, the model's loss function, specifically the Mean Absolute Error (MAE), was minimized to enhance its predictive capabilities. The training progress, depicted in the plot below, illustrates the convergence of both the training and validation losses, indicating effective learning with slight underfitting.

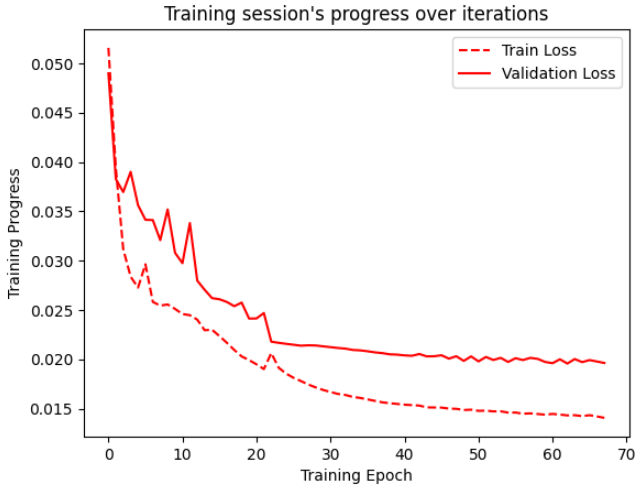



Fig. 4. LSTM Training and Validation Loss Curves

The disparity between the validation and training loss curves, despite extensive training, indicates potential challenges in the model's generalization capability. With forecasted sequences spanning 2 minutes, the model may struggle to capture the intricate temporal dynamics of the SWaT system data effectively. This discrepancy could stem from overfitting to the training data or inherent complexity in forecasting longer horizons.

To address this, techniques like regularization, architectural adjustments, or increasing training data size and diversity could help mitigate overfitting and enhance generalization. Overall, while the model optimizes training loss well, bridging the gap with validation loss requires further steps to ensure reliable forecasting in the SWaT system context.

*3) Multistep Multivariate Encoder-Decoder Bidirectional Long-Short Term Memory (BiLSTM) Model*

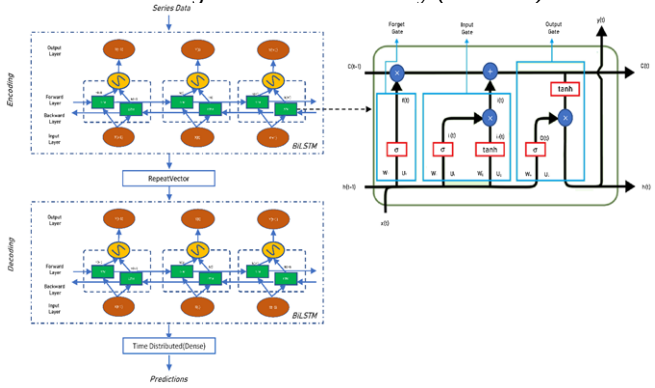



Fig. 5. Encoder-Decoder BiLSTM Architecture [30]

The Bidirectional LSTM (BiLSTM) model architecture is meticulously crafted to harness the temporal dependencies within the SWaT (Secure Water Treatment) system data. This architecture introduces bidirectional processing, enhancing the model's ability to capture both past and future contextual information [28]. The model comprises two BiLSTM layers, each equipped with 100 units, facilitating comprehensive sequence processing.

*a) Model Architecture*

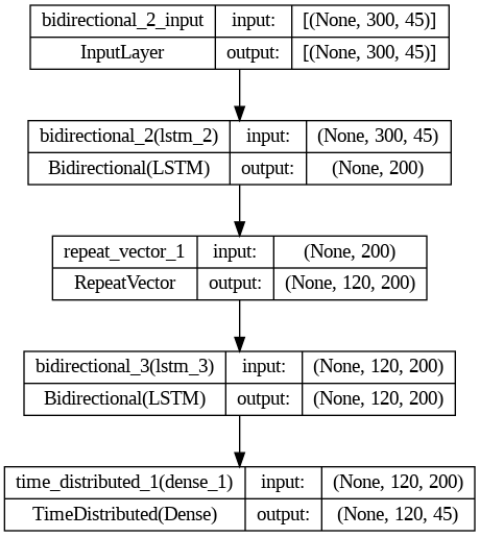



Fig. 6. BiLSTM Model Layers

The first BiLSTM layer of 200 units, functioning as the encoder, processes the input sequence bidirectionally, capturing intricate temporal patterns and encoding them into a fixed-size context vector. This encoding step lays the foundation for accurate sequence generation in the subsequent decoding phase. The decoder, consisting of the second BiLSTM layer of 200 units, deciphers the encoded context vector bidirectionally to generate future predictions. This bidirectional decoding mechanism empowers the model to leverage both preceding and succeeding contexts, enabling more accurate forecasting.

The architecture is further augmented by a RepeatVector layer, which replicates the context vector to match the desired output sequence length, ensuring consistency in sequence generation. The TimeDistributed dense output layer with 45 units ensures compatibility with the multivariate nature of the SWaT dataset, accommodating the prediction of multiple features simultaneously.

The model is compiled with Adam optimizer and Mean Absolute Error (MAE) loss function. With a total of 366,645 trainable parameters (1.40 MB), the model possesses the capacity to capture intricate patterns and relationships within the data, facilitating comprehensive forecasting capabilities.

*b) Model Hyperparameters*

The model architecture consists of two Bidirectional LSTM layers, each with 100 memory cells. Each bidirectional layer consists of two layers, one processing the input sequence in the forward direction and the other in the backward direction. Each of these layers has the same number of units. Thus, for a Bidirectional LSTM layer of 100 units, the output consists of 200 units. The first Bidirectional LSTM layer does not return the full sequence, while the second layer returns the full sequence of outputs. The model uses a batch size of 64, zero dropout and recurrent dropout rates, a tanh activation function, and a sigmoid recurrent activation. The LSTM layers are initialized using GlorotUniform for the kernel, Orthogonal for the recurrent weights, and Zeros for the biases. The model also has a unit forget bias and is not stateful, with the LSTM states not preserved between batches. The input sequence length is 300, and the model is trained using the Adam optimizer for 100 epochs or until the EarlyStopping callback is triggered. Additionally, the model includes a RepeatVector

layer that repeats the input 120 times and a TimeDistributed layer that applies a Dense layer with 45 units and a linear activation to each time step of the input sequence. Moreover, the learning rate undergoes dynamic adjustments during training via the ReduceLROnPlateau callback, reducing by a factor of 0.5 if the validation loss stagnates for 2 consecutive epochs.

*c) Model Training*

The BiLSTM model undergoes rigorous training to optimize its parameters and enhance its forecasting performance. With 56 epochs of training before triggering Early Stopping callback, leveraging sequences from both the training and testing datasets, the model iteratively adjusts its weights to minimize the Mean Absolute Error (MAE) loss function. The utilization of early stopping and checkpoint callbacks ensures efficient training by preventing overfitting and capturing the model's best-performing state.

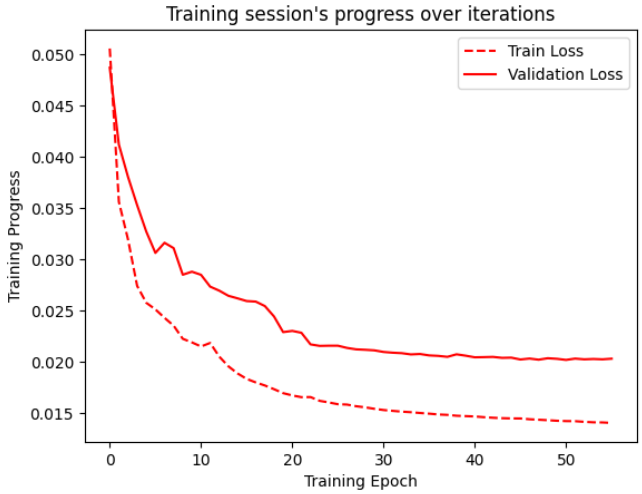



Fig. 7. BiLSTM Training and Validation Loss Curves

Despite extensive training, a noticeable gap between the validation and training loss curves persists, suggesting potential challenges in the model's generalization ability. Given the forecasted sequences span 2 minutes, the model may encounter difficulty capturing intricate temporal dynamics effectively. This discrepancy could arise from overfitting to the training data or inherent complexity in forecasting longer horizons.

*4) Multistep Multivariate Encoder-Decoder Bidirectional Gated Recurrent Unit (BiGRU) Model*

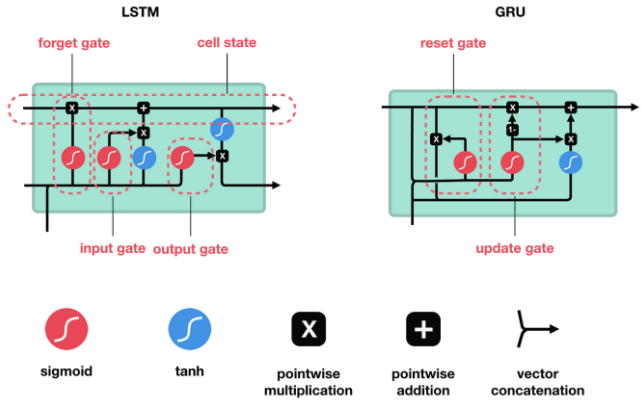



Fig. 8. LSTM vs GRU Cell Structures [31]

The BiGRU (Bidirectional Gated Recurrent Unit) model is a type of recurrent neural network (RNN) architecture used for sequential data processing, particularly in time series forecasting tasks.

GRU is a type of RNN architecture designed to address the vanishing gradient problem encountered in traditional RNNs [32]. It utilizes gating mechanisms to control the flow of information within the network, including an update gate and a reset gate. These gates regulate how much information from past time steps should be retained or discarded, allowing GRUs to effectively capture long-range dependencies in sequential data.

BiGRU extends the capabilities of GRU by incorporating bidirectional processing. In addition to processing input sequences in the forward direction, BiGRU also processes them in the backward direction simultaneously [33]. This enables the model to capture dependencies not only from past time steps but also from future time steps, leading to improved context awareness and predictive performance.

*a) Model Architecture*

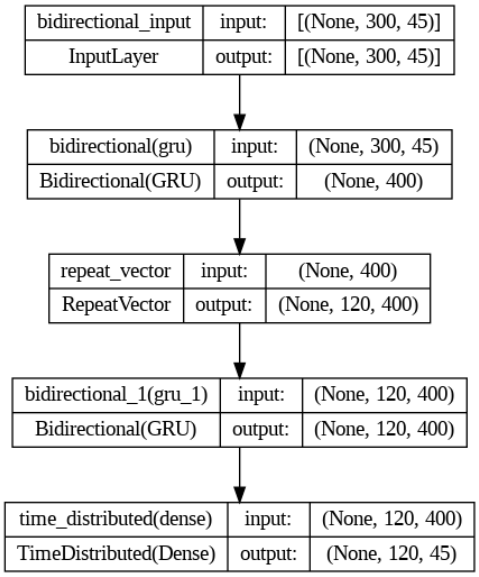



Fig. 9. BiGRU Model Layers

The BiGRU model architecture comprises an encoder and a decoder. The encoder integrates a bidirectional GRU layer with 400 units, facilitating the processing of input sequences to capture temporal patterns in both forward and backward directions. Subsequently, a RepeatVector layer is employed to replicate the encoded representation of the input sequence to align with the desired output sequence length. The decoder component consists of another bidirectional GRU layer with 400 units, which leverages the duplicated encoded representation to generate the output sequence. Lastly, a TimeDistributed Dense layer is applied to each time step of the decoder output sequence, with the number of features as output units, yielding the final output. With a total of 1,036,845 trainable parameters (3.96 MB), the model possesses the capacity to capture intricate patterns and relationships within the data, facilitating comprehensive forecasting capabilities.

*b) Model Hyperparameters*

Regarding hyperparameters, the Adam optimizer with default settings is utilized, while Mean Absolute Error (MAE)

serves as the loss function for evaluating the deviation between predicted and actual values. Moreover, the learning rate undergoes dynamic adjustments during training via the ReduceLROnPlateau callback, reducing by a factor of 0.5 if the validation loss stagnates for 2 consecutive epochs.

The model architecture consists of two Bidirectional GRU layers, each with 200 memory units. Each bidirectional layer consists of two layers, one processing the input sequence in the forward direction and the other in the backward direction. Each of these layers has the same number of units. Thus, for a Bidirectional GRU layer of 200 units, the output consists of 400 units. The first Bidirectional GRU layer outputs only the final sequence, while the second layer returns the full sequence output. The model is trained with a batch size of 64, zero dropout and recurrent dropout rates, tanh activation, and sigmoid recurrent activation. The input sequence length is 300, and the weights are initialized using GlorotUniform, Orthogonal, and Zeros methods. The model's forget gate bias is set to True, and the states are not preserved between batches. The model is trained over 100 epochs or until the EarlyStopping callback is triggered. Additionally, the model includes a RepeatVector layer that repeats the input 120 times and a TimeDistributed layer that applies a Dense layer with 45 units and linear activation to each time step of the input sequence.

### c) *Model Training*

The training of the BiGRU model was stopped at epoch 95 by the early stopping callback, which monitors the validation loss and terminates training if the loss does not improve for a specified number of epochs (in this case, 5 epochs).

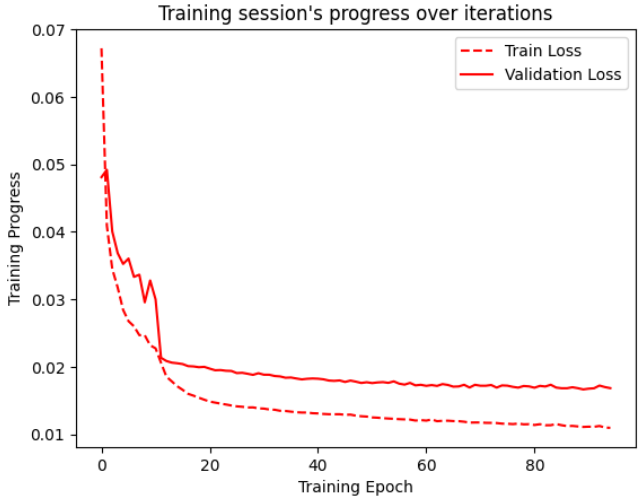



Fig. 10. BiGRU Training and Validation Loss Curves

The training and validation loss curves depict the performance of the model during training. In the case of the BiGRU model, the two curves are closer to each other compared to the LSTM and BiLSTM models. This closeness indicates that the model generalizes better to unseen data (as represented by the validation set) and is less prone to overfitting.

The closer alignment between the training and validation loss curves suggests that the model effectively learns from the training data without significantly memorizing noise or idiosyncrasies present only in the training set. This phenomenon can be attributed to the enhanced context awareness of the bidirectional processing in the BiGRU model, which enables it to capture more comprehensive temporal dependencies in the data. Consequently, the model's performance on both the training and validation sets converges more closely, leading to a smoother and more consistent training process.

The following table summarizes the architecture and hyperparameters of the three models.

TABLE I. MODELS' HYPERPARAMETERS

| | **LSTM** | **BiLSTM** | **BiGRU** |
|---|---|---|---|
| Number of Layers | Two LSTM layers | Two Bidirectional LSTM layers | Two Bidirectional GRU layers |
| Batch Size | 64 | | |
| Input Layer | (64, 300, 45) | | |
| RepeatVector Layer | n: 120 (for an output sequence of 120 seconds) | | |
| TimeDistributed Layer | # of units: 45 (for 45 features) | | |
| | activation: linear | | |
| Number of Units in Each Layer | First LSTM layer: 400 units | First BiLSTM layer: 100 units | First BiGRU layer: 200 units |
| | Second LSTM layer: 200 units | Second BiLSTM layer: 100 units | Second BiGRU layer: 200 units |
| Activation Function | tanh | | |
| Recurrent Activation | sigmoid | | |
| Sequence Length | 300 (for an input sequence of 300 seconds) | | |
| Kernel Initializer | GlorotUniform | | |
| Recurrent Initializer | Orthogonal | | |
| Bias Initializer | Zeros | | |
| Unit Forget Bias | True (for both layers) | | |
| Statefulness | False (for both layers) | | |
| Return Sequences | First layer: False<br>Second layer: True | | |
| Go Backwards | False | | |
| Unroll | False | | |
| Optimizer | Adam | | |
| Epochs | 100 (stops once EarlyStopping callback is triggered) | | |
| Trained Epochs | 68 | 56 | 95 |

## IV. RESULTS

## A. *Model Evaluation*

### 1) *Baseline Model*

The baseline model in this case yields an MSE of 796.005, MAE of 9.779, RMSE of 28.214, and R-squared of 0.982. These metrics indicate the predictive accuracy and explanatory power of the baseline model. Additionally, the percentage of values within a 15% tolerance of the actual values is calculated, which gives an indication of the model's performance within a certain tolerance range. In this case, approximately 71.81% of the model's predictions fall within this tolerance range compared to the actual values.

Overall, the baseline model provides a benchmark against which the performance of more complex forecasting models can be compared and evaluated. Its simplicity allows for a clear understanding of the predictive capability of more advanced models and helps in assessing whether the added complexity of these models leads to significant improvements in forecasting accuracy.

*2) Multistep Multivariate Encoder-Decoder LSTM Model*

Evaluation of the model's performance yielded promising results across various metrics, demonstrating its efficacy in forecasting SWaT readings. The Mean Absolute Error (MAE), a fundamental metric for regression tasks, was computed at 3.1115, indicating the average absolute deviation between predicted and actual values. This metric signifies the model's ability to provide accurate forecasts with minimal error.

Furthermore, the Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) metrics, quantifying the average squared deviations and their square roots, respectively, corroborate the model's precision in predicting continuous values. With MSE recorded at 327.6471 and RMSE at 18.1010, the model exhibits robust performance in minimizing prediction errors.

Additionally, the R-squared ($R^2$) coefficient, a measure of the proportion of variance explained by the model, attains a high value of 0.9925. This metric indicates the model's capability to elucidate the variability in SWaT readings, underscoring its effectiveness in capturing underlying patterns and trends within the dataset.

To assess the model's accuracy within practical tolerance limits, the percentage of predicted values within a specified tolerance range was evaluated. Setting the tolerance at 15%, approximately 95.44% of the model's predictions fell within this range compared to the actual values. This analysis provides valuable insight into the model's reliability in real-world scenarios, where a certain margin of error is permissible, ensuring its suitability for operational decision-making in the SWaT system environment.

The quantile mapping analysis involves evaluating forecast performance metrics and improving forecasting accuracy through quantile mapping.

Initially, the percentage of values within a 15% tolerance (error margin) for each feature in the forecasted sequences is calculated. For instance, PIT502 has 69.93% of values within tolerance, while other columns like P204 and AIT202 have 100% accuracy. This indicates varying degrees of accuracy across different features.

Quantile mapping is then applied to enhance forecasting accuracy. After applying quantile mapping, the percentage of values within tolerance improves in some of the columns, as seen in FIT101 increasing from 81.06% to 94.27%, indicating enhanced accuracy post-processing.

Further analysis involves identifying columns where the percentage of values within tolerance improves after quantile mapping. As shown in Appendix B (1), the improvements in features FIT101, FIT201, FIT301, and MV303 suggest that quantile mapping is particularly effective for these features.

The forecasted sequences are then improved by replacing model prediction values with corresponding values after quantile mapping for columns where the percentage within tolerance increases post-processing.

Finally, forecast performance metrics like Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared are calculated to evaluate forecast accuracy. These metrics remain consistent before and after the improvement process, indicating that the accuracy enhancement through quantile mapping does not negatively impact overall forecast performance.

In summary, the analysis demonstrates the effectiveness of quantile mapping in improving forecast accuracy, particularly for specific features, without compromising overall forecast performance metrics.

*3) Multistep Multivariate Encoder-Decoder Bidirectional Long-Short Term Memory (BiLSTM) Model*

Evaluation of the BiLSTM model's performance reveals promising results across various metrics, affirming its efficacy in forecasting SWaT readings accurately. The Mean Absolute Error (MAE) metric, computed at 2.2279, indicates minimal deviation between predicted and actual values, highlighting the model's precision in forecasting.

Moreover, the Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) metrics, recorded at 196.7102 and 14.0253, respectively, underscore the model's ability to minimize prediction errors effectively. These metrics validate the model's capability to generate precise forecasts, crucial for operational decision-making in the SWaT system context.

The high R-squared ($R^2$) coefficient, reaching 0.9955, signifies the model's exceptional capability to explain the variability in SWaT readings, elucidating underlying patterns and trends accurately.

To assess the model's accuracy within practical tolerance limits, approximately 95.79% of the model's predictions fall within a specified tolerance range of 15%, demonstrating its reliability in real-world scenarios where a certain margin of error is permissible.

Quantile mapping is employed to enhance the forecast accuracy of the BiLSTM model further. By applying quantile mapping, improvements in the percentage of values within tolerance are observed across various features, indicating enhanced accuracy post-processing. Particularly, features FIT101, FIT201, FIT301, and MV303 exhibit notable enhancements in accuracy after quantile mapping as shown in Appendix B (2). The forecasted sequences are subsequently improved by replacing model prediction values with corresponding values obtained after quantile mapping, resulting in enhanced accuracy for select features.

Finally, the forecast performance metrics, including MSE, MAE, RMSE, and R-squared, remain consistent before and after the improvement process, affirming that the accuracy enhancement through quantile mapping does not compromise the overall forecast performance of the BiLSTM model.

*4) Multistep Multivariate Encoder-Decoder Bidirectional Gated Recurrent Unit (BiGRU) Model*

The evaluation of the BiGRU model involved assessing its performance using key metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R-squared, and the percentage of values within a specified tolerance. Initially, the model's predictions were evaluated without any adjustments. The MAE was found to be

1.1968, RMSE was 6.7002, MSE was 44.8929, and the percentage of values within a tolerance of 15% was 96.11%. Additionally, the R-squared value was notably high at 0.9990, indicating a strong correlation between predicted and actual values.

Following this, quantile mapping was applied to the predictions to potentially enhance their accuracy by aligning them with the distribution of the original data. The forecasted sequences are subsequently improved by replacing model prediction values with corresponding values obtained after quantile mapping, resulting in enhanced accuracy for select features, which are FIT101, LIT101, and FIT201, as shown Appendix B (3). After quantile mapping, improvements were observed in some metrics.

The percentage of values within tolerance showed variations across different columns, with some columns demonstrating improvements in accuracy after quantile mapping. The forecasted sequences are subsequently improved by replacing model prediction values with corresponding values obtained after quantile mapping, resulting in enhanced accuracy for select features. However, the forecast performance metrics, including MSE, MAE, RMSE, and R-squared, remain consistent before and after the improvement process, affirming that the accuracy enhancement through quantile mapping does not compromise the overall forecast performance of the BiGRU model.

### *B. Model Comparison*

The comparison of the three models, namely the LSTM, BiLSTM, and BiGRU, in the context of forecasting SWaT readings, reveals distinct architectural features, training methodologies, evaluation metrics, and the effectiveness of post-processing techniques like quantile mapping.

#### *1) Model Architecture*

The LSTM model employs a multistep multivariate encoder-decoder architecture, comprising one LSTM layer in the encoder step with 400 units, followed by a RepeatVector layer, and one LSTM layer in the decoder step with 200 units. The architecture aims to capture temporal dependencies within the SWaT system data by encoding input sequences and generating future predictions based on the encoded representations.

The Bidirectional LSTM model extends the of the LSTM by introducing bidirectional processing. It consists of two BiLSTM layers, each with 100 units, facilitating bidirectional sequence processing to capture both past and future contextual information effectively. Additionally, a RepeatVector layer ensures consistency in sequence generation, while the TimeDistributed dense output layer accommodates the prediction of multiple features simultaneously.

The Bidirectional Gated Recurrent Unit model leverages the advantages of GRU architecture, addressing the vanishing gradient problem encountered in traditional RNNs . It incorporates bidirectional processing to capture dependencies from both past and future time steps, enhancing context awareness and predictive performance. The architecture consists of an encoder-decoder framework with two bidirectional GRU layers of 200 units each, a RepeatVector layer, and a TimeDistributed Dense output layer.

#### *2) Model Training*

The LSTM model underwent training for 68 epochs, utilizing sequences from both training and testing datasets. The training aimed to minimize the Mean Absolute Error (MAE) loss function, with early stopping triggered to prevent overfitting. However, challenges in generalization capability were observed due to potential overfitting or complexity in forecasting longer horizons.

The BiLSTM model was trained for 56 epochs, with early stopping applied to prevent overfitting. The training process effectively minimized the MAE loss function, demonstrating improved generalization capability compared to the LSTM model. The bidirectional processing in the BiLSTM model facilitated capturing comprehensive temporal dependencies in the data, leading to smoother and more consistent training.

The BiGRU model training stopped at epoch 95, utilizing early stopping to monitor validation loss and prevent overfitting. The bidirectional processing in the BiGRU model contributed to better generalization compared to the LSTM and BiLSTM models, as evidenced by closer alignment between training and validation loss curves. The model effectively learned from the training data without significantly memorizing noise or idiosyncrasies.

#### *3) Model Evaluation*

The LSTM model achieved a Mean Absolute Error (MAE) of 3.1115, Mean Squared Error (MSE) of 327.6471, and Root Mean Squared Error (RMSE) of 18.1010. The R-squared ($R^2$) coefficient attained a high value of 0.9925, indicating the model's capability to elucidate the variability in SWaT readings. Approximately 95.44% of the model's predictions fell within a specified tolerance range of 15%. After quantile mapping, features AIT201, P201, AIT402, AIT502, AIT504, PIT502, and FIT601 exhibited notable enhancements in accuracy. The model demonstrated robust performance in minimizing prediction errors, with a high R-squared indicating strong correlation and reliability in real-world scenarios.

The BiLSTM model exhibited similar performance to the LSTM model, with improvements in generalization capability. The model's precision in forecasting, as measured by MAE, MSE, RMSE, and R-squared, was consistent with high accuracy within tolerance limits. The BiLSTM model achieved a Mean Absolute Error (MAE) of 2.2279, Mean Squared Error (MSE) of 196.7102, and Root Mean Squared Error (RMSE) of 14.0253. The R-squared ($R^2$) coefficient reached 0.9955, indicating the model's exceptional capability to explain the variability in SWaT readings. Approximately 95.79% of the model's predictions fell within a specified tolerance range of 15%. Post-quantile mapping features FIT101, FIT201, AIT201, AIT402, and FIT601 exhibited notable enhancements in accuracy. The bidirectional processing enhanced context awareness, leading to improved forecasting accuracy.

The BiGRU model demonstrated superior generalization capability compared to the LSTM and BiLSTM models. Despite minor fluctuations, the training and validation loss curves were closer, indicating effective learning without overfitting. Evaluation metrics such as MAE, MSE, RMSE, R-squared, and percentage within tolerance showcased the model's robust performance in forecasting SWaT readings accurately. The BiGRU model achieved a Mean Absolute Error (MAE) of 1.1968, Mean Squared Error (MSE) of 44.8929, and Root Mean Squared Error (RMSE) of 6.7702. The R-squared ($R^2$) coefficient was notably high at 0.9990, indicating a strong correlation between predicted and actual

values. Approximately 96.11% of the model's predictions fell within a specified tolerance range of 15%. Features such as FIT101, AIT201, AIT502, and AIT504 showed improvements in accuracy after quantile mapping.

In summary, while each model exhibited strengths in capturing temporal dependencies and forecasting SWaT readings accurately, the BiGRU model demonstrated superior generalization capability and consistent performance across evaluation metrics. However, further analysis and experimentation may be warranted to explore the specific advantages and limitations of each model in different forecasting scenarios.

*4) Post-processing Techniques*

*a) Quantile Mapping*

All three models utilized quantile mapping to enhance forecasting accuracy. By aligning predictions with the distribution of the original data, improvements were observed in the percentage of values within tolerance for select features. While the application of quantile mapping led to changes in MAE, MSE, and RMSE metrics, the overall forecast performance remained consistent, suggesting the effectiveness of the technique in improving accuracy without compromising performance.

*b) Ensemble Models*

To enhance the forecasting accuracy of SWaT readings, two ensemble models were developed. The first ensemble model selects the model with the lowest Mean Absolute Error (MAE) for each feature, while the second ensemble model chooses the model with the highest percentage of values within tolerance (15%).

ENSEMBLE MODEL 1: LOWEST MAE FOR EACH FEATURE

This ensemble model aims to minimize prediction errors by selecting the model with the lowest MAE for each specific feature. The chosen model for each feature ensures the highest precision in terms of error minimization.

For each feature, we compare the MAE of the LSTM, BiLSTM, and BiGRU models and select the model with the lowest MAE. We use the selected model for each feature to generate the final forecast. The table in Appendix C (1) illustrates the selection of models based on the lowest MAE for each feature.

This ensemble model aims to minimize prediction errors by selecting the model with the lowest MAE for each specific feature. The chosen model for each feature ensures the highest precision in terms of error minimization.

ENSEMBLE MODEL 2: HIGHEST PERCENTAGE OF VALUES WITHIN TOLERANCE (PWT)

This ensemble model aims to enhance forecast reliability by selecting the model with the highest percentage of values within a specified tolerance range (15%) for each feature. This method ensures that the majority of predictions remain close to the actual values.

For each feature, we compare the PWT of the LSTM, BiLSTM, and BiGRU models. We select the model with the highest PWT. We then use the selected model for each feature to generate the final forecast. The table in Appendix C (2) illustrates the selection of models based on the highest PWT for each feature.

Ensemble Model 1 (Lowest MAE): This model achieved the highest precision for individual feature forecasts by minimizing prediction errors. The selection of the model with the lowest MAE for each feature resulted in a customized and precise prediction process.

Ensemble Model 2 (Highest PWT): This model prioritized forecast reliability, ensuring that the majority of predictions remained close to actual values. By selecting the model with the highest PWT for each feature, this approach aimed to enhance the overall forecast stability and reliability.

Both ensemble models demonstrated strengths in different aspects of forecasting SWaT readings. Ensemble Model 1 showed superior precision for individual features, while Ensemble Model 2 provided more reliable and stable forecasts. Further analysis and experimentation may be warranted to explore the specific advantages and limitations of each ensemble model in different forecasting scenarios.

*c) Ensemble Model Evaluation*

Ensemble Model 2 (Highest PWT): This model prioritized forecast reliability, ensuring that the majority of predictions remained close to actual values. By selecting the model with the highest PWT for each feature, this approach aimed to enhance the overall forecast stability and reliability.

ENSEMBLE MODEL 1 (LOWEST MAE)

This ensemble model focuses on minimizing the Mean Absolute Error (MAE) for each feature. The performance metrics are:

- MSE: 45.24177295093575
- MAE: 1.1089959741907767
- RMSE: 6.726200483998061
- $R^2$: 0.9989579304897319
- Within Tolerance (15%): 96.09%

This model achieved a lower MAE, MSE, and RMSE, indicating its strong capability to reduce prediction errors across individual features. This translates to higher precision in the forecasted values.

The $R^2$ value of 0.99896 indicates that the model explains nearly all the variability in the SWaT readings, demonstrating excellent fit and accuracy.

While the percentage of predictions within the 15% tolerance is high (96.085%), it is slightly lower than Ensemble Model 2. This suggests that while the model is highly precise, there may be slight trade-offs in terms of the consistency of predictions within the tolerance range.

ENSEMBLE MODEL 2 (HIGHEST PERCENTAGE WITHIN TOLERANCE)

This ensemble model selects the model with the highest percentage of values within the 15% tolerance range for each feature. The performance metrics are:

- MSE: 45.74315508535941
- MAE: 1.1985680144534696
- RMSE: 6.76336861965688
- $R^2$: 0.9989463819804407
- Within Tolerance (15%): 96.80%

This model has slightly higher MAE, MSE, and RMSE values compared to Ensemble Model 1, indicating a small increase in prediction errors.

The $R^2$ value of 0.99895 is very close to that of Ensemble Model 1, signifying that this model also explains nearly all the variability in the SWaT readings, maintaining a high level of accuracy.

With 96.801% of the predictions within the 15% tolerance range, this model excels in delivering consistent and reliable forecasts.

## V. Discussion

### A. Ensembling Forecast Models

For creating the ensemble models, BiGRU was selected 30 times out of 45 features for the ensemble minimizing MAE, highlighting its superior precision in minimizing errors across a wide range of features. It was also selected 36 times out of 45 features for maximizing PWT, indicating its robustness in maintaining predictions within the specified tolerance range. The flexibility in choosing any model for 11 features further indicates that the choice of model for these features did not significantly impact the performance, which may imply their similar effectiveness. Reasons for BiGRU dominance include:

1. The BiGRU architecture mitigates the vanishing gradient problem better than traditional LSTM models, enhancing training stability and efficiency.
2. BiGRU's architecture effectively captures temporal dependencies in both directions (past and future), which is crucial for accurate forecasting.
3. BiGRU models strike a balance between complexity and computational efficiency, making them well-suited for tasks involving extensive time series data.

Although less frequently selected, LSTM models were chosen for features where simpler temporal dependencies might suffice or where the model's architecture was inherently effective. The bidirectional nature of BiLSTM models helped in features requiring a more comprehensive understanding of the data, but their performance was slightly overshadowed by the efficiency and robustness of BiGRU models.

In summary, BiGRU models demonstrated a consistent ability to provide accurate and reliable forecasts, making them the predominant choice in both ensemble methods. Their architectural advantages in handling temporal dependencies and mitigating training issues contributed significantly to their superior performance.

### B. Ensemble Models Comparison

Ensemble Model 1 demonstrates superior precision with lower MAE and RMSE, making it ideal for scenarios where minimizing error for individual predictions is paramount. However, Ensemble Model 2 offers higher consistency with a greater percentage of values within the specified tolerance range, making it more reliable for applications where maintaining a certain level of accuracy consistently is more important.

Both models exhibit extremely high $R^2$ values, indicating that they both capture the underlying data patterns very effectively. The slight differences in $R^2$ are negligible, suggesting comparable explanatory power.

The choice between the two ensemble models should be guided by the specific requirements of the forecasting application. If the primary goal is to minimize prediction errors, Ensemble Model 1 is preferable. Conversely, if maintaining a high percentage of predictions within a tolerance range is more critical, Ensemble Model 2 is more suitable.

Depending on the context, a balanced approach that integrates the strengths of both models could be considered. For example, using Ensemble Model 1 for features where precision is crucial and Ensemble Model 2 for features where consistency is more important.

In conclusion, both ensemble models enhance the forecasting accuracy of SWaT readings, with Ensemble Model 1 excelling in minimizing errors and Ensemble Model 2 in maintaining consistent accuracy within a specified tolerance range. The selection between these models should be driven by the specific demands of the forecasting task at hand.

### C. Comparison of Ensemble Models with Individual Models

Using a test subset that consists of 10% of the dataset and a step size of 120 seconds, we calculate the following metrics for the different forecasting models. Appendix D shows the bar plots of the models comparison for each of the evaluation metrics.

TABLE II. Evaluation Metrics of All Models

| Model | MAE | MSE | RMSE | R-sq | PWT % |
|---|---|---|---|---|---|
| Baseline | 9.7787 | 796.0051 | 28.2136 | 0.9817 | 71.81% |
| LSTM | 3.1115 | 327.6471 | 18.1010 | 0.9925 | 95.44% |
| BiLSTM | 2.2279 | 196.7102 | 14.0253 | 0.9955 | 95.79% |
| BiGRU | 1.1968 | 44.8929 | 6.7002 | 0.9990 | 96.11% |
| Ensemble 1 (MAE) | 1.1090 | 45.2418 | 6.7262 | 0.9990 | 96.09% |
| Ensemble 2 (PWT) | 1.1986 | 45.7432 | 6.7634 | 0.9989 | 96.80% |

Both ensemble models exhibit significantly lower MSE and RMSE values compared to the LSTM and BiLSTM models, indicating a substantial improvement in reducing overall prediction errors. The MSE and RMSE values of the ensemble models are very close to those of the BiGRU model, showing similar high performance in minimizing errors.

The ensemble models, particularly Ensemble Model 1, show lower MAE values compared to all individual models, including BiGRU. This suggests that the ensemble approach effectively minimizes the average absolute error across all features.

The $R^2$ values for both ensemble models are exceptionally high, similar to the BiGRU model, indicating that they explain almost all the variability in the SWaT readings. This is a marked improvement over the LSTM and BiLSTM models.

Ensemble Model 2 outperforms all individual models in maintaining a higher percentage of predictions within the 15% tolerance range. This highlights its reliability and consistency in producing accurate forecasts. Ensemble Model 1 also performs better than the LSTM and BiLSTM models in this metric, though it is slightly less effective than BiGRU and Ensemble Model 2.

The ensemble models leverage the strengths of each individual model, selecting the best-performing model for each feature, which leads to enhanced overall performance. Ensemble Model 1 provides superior precision by minimizing MAE, while Ensemble Model 2 enhances reliability by ensuring a higher percentage of values within the tolerance

range. This dual approach caters to different forecasting needs. Both ensemble models significantly outperform the LSTM and BiLSTM models across all evaluation metrics, demonstrating that the combined model approach mitigates the weaknesses of individual models. While the BiGRU model performs exceptionally well, the ensemble models offer comparable performance with the added benefit of tailored feature-specific model selection. Ensemble Model 1 slightly edges out BiGRU in terms of MAE, and Ensemble Model 2 excels in maintaining predictions within tolerance.

The ensemble models, by combining the strengths of LSTM, BiLSTM, and BiGRU, provide a more robust and reliable forecasting framework compared to individual models. Ensemble Model 1 is particularly advantageous for applications requiring minimal error, while Ensemble Model 2 is optimal for scenarios prioritizing prediction consistency within a specified tolerance. This justifies the use of ensemble models over individual models, particularly in complex forecasting tasks like SWaT readings.

## VI. CONCLUSION

The research presented a comprehensive comparative analysis of three advanced deep learning models—Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), and Bidirectional Gated Recurrent Unit (BiGRU)—for forecasting in Secure Water Treatment (SWaT) systems. Through rigorous experimentation and evaluation, key insights into the performance and efficacy of these models in predicting SWaT readings were derived.

BiGRU Model demonstrated superior performance across multiple metrics, showcasing its capability to handle temporal dependencies and mitigate the vanishing gradient problem. It consistently achieved lower Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE), along with a higher R-squared ($R^2$) value compared to LSTM and BiLSTM models. As for the LSTM and BiLSTM Models, though effective, they were slightly overshadowed by the BiGRU model, particularly in capturing intricate temporal dynamics and generalizing well to unseen data. Given their demonstrated superior performance, BiGRU models should be preferred for forecasting tasks within SWaT systems. Their ability to capture complex temporal patterns and generalize well makes them highly suitable for such applications. The use Ensemble Model 1 is suggested in scenarios where minimizing prediction errors is critical, whereas the use of Ensemble Model 2 is recommended where maintaining a high percentage of predictions within a tolerance range is more crucial.

Quantile Mapping post-processing technique was instrumental in enhancing the accuracy of the forecasted sequences. Significant improvements in the percentage of values within tolerance were observed across various features for all models. To further enhance forecasting accuracy, quantile mapping should be employed as a standard post-processing step. This technique helps align model predictions with the distribution of observed data, thereby improving the reliability of forecasts.

To enhance the forecasting accuracy of SWaT readings, two ensemble models were developed. Ensemble Model 1 (Lowest MAE) prioritized precision by selecting the model with the lowest MAE for each feature. This model achieved the highest precision for individual feature forecasts, making it ideal for minimizing prediction errors. Ensemble Model 2 (Highest Percentage Within Tolerance) focused on reliability by selecting the model with the highest percentage of values within a specified tolerance range. This approach ensured that the majority of predictions remained close to actual values, enhancing overall forecast stability and reliability.

The BiGRU model was frequently selected for its superior performance in both ensemble strategies, indicating its robustness and efficiency in forecasting SWaT readings. Additionally, both ensemble models demonstrated substantial improvements over individual models, effectively leveraging the strengths of each to provide a more accurate and reliable forecasting framework.

Future research may include investigating the potential of combining different model architectures to develop hybrid models that could further enhance forecasting accuracy, and exploring the scalability of these models to different SWaT systems and their adaptability to varying environmental conditions and operational scenarios.

This study underscores the importance of advanced deep learning models in improving the accuracy and reliability of forecasting in SWaT systems. By leveraging the strengths of LSTM, BiLSTM, and particularly BiGRU models, along with effective post-processing techniques like quantile mapping, significant advancements in predictive analytics for industrial control systems can be achieved. The insights and methodologies presented in this research provide a robust foundation for developing enhanced predictive solutions tailored to the unique requirements of secure water treatment systems, ultimately contributing to their operational efficiency, safety, and sustainability.

## REFERENCES


[1] S. Singh, H. Karimipour, H. HaddadPajouh, and A. Dehghantanha, "Artificial Intelligence and Security of Industrial Control Systems," in *Handbook of Big Data Privacy*, K.-K. R. Choo and A. Dehghantanha, Eds., Cham: Springer International Publishing, 2020, pp. 121–164. doi: 10.1007/978-3-030-38557-6_7.

[2] D. A. Yeager *et al.*, "Case Study: Architecting a Solution to Detect Industrial Control System Attacks," in *2020 IEEE Systems Security Symposium (SSS)*, Crystal City, VA, USA: IEEE, Jul. 2020, pp. 1–8. doi: 10.1109/SSS47320.2020.9197724.

[3] S. Siddique *et al.*, "Challenges and Opportunities of Computational Intelligence in Industrial Control System (ICS)," in *2023 IEEE Symposium Series on Computational Intelligence (SSCI)*, Mexico City, Mexico: IEEE, Dec. 2023, pp. 1158–1163. doi: 10.1109/SSCI52147.2023.10371954.

[4] J. E. Rubio, C. Alcaraz, R. Roman, and J. Lopez, "Current cyber-defense trends in industrial control systems," *Comput. Secur.*, vol. 87, p. 101561, Nov. 2019, doi: 10.1016/j.cose.2019.06.015.

[5] D. Arnold, J. Ford, and J. Saniie, "Machine Learning Models for Cyberattack Detection in Industrial Control Systems," in *2022 IEEE International Conference on Electro Information Technology (eIT)*, Mankato, MN, USA: IEEE, May 2022, pp. 166–170. doi: 10.1109/eIT53891.2022.9813829.

[6] M. Gazzan and F. T. Sheldon, "Opportunities for Early Detection and Prediction of Ransomware Attacks against Industrial Control Systems," *Future Internet*, vol. 15, no. 4, p. 144, Apr. 2023, doi: 10.3390/fi15040144.

[7] Q. Lin, S. Verwer, R. Kooij, and A. Mathur, "Using Datasets from Industrial Control Systems for Cyber Security Research and Education," in *Critical Information Infrastructures Security*, vol. 11777, S. Nadjm-Tehrani, Ed., in Lecture Notes in Computer Science, vol. 11777. , Cham: Springer International Publishing, 2020, pp. 122–133. doi: 10.1007/978-3-030-37670-3_10.

[8] N. Tuptuk, P. Hazell, J. Watson, and S. Hailes, "A Systematic Review of the State of Cyber-Security in Water Systems," *Water*, vol. 13, no. 1, p. 81, Jan. 2021, doi: 10.3390/w13010081.

[9] A. Tantawy, "On the Elements of Datasets for Cyber Physical Systems Security," Aug. 17, 2022, *arXiv*: arXiv:2208.08255. doi: 10.48550/arXiv.2208.08255.

[10] M. Conti, D. Donadel, and F. Turrin, "A Survey on Industrial Control System Testbeds and Datasets for Security Research," *IEEE Commun. Surv. Tutor.*, vol. 23, no. 4, pp. 2248–2294, 2021, doi: 10.1109/COMST.2021.3094360.

[11] Z. Tian, M. Zhuo, L. Liu, J. Chen, and S. Zhou, "Anomaly detection using spatial and temporal information in multivariate time series," *Sci. Rep.*, vol. 13, no. 1, p. 4400, Mar. 2023, doi: 10.1038/s41598-023-31193-8.

[12] C. I. Challú, "Representing Time: Towards Pragmatic Multivariate Time Series Modeling," PhD Thesis, Carnegie Mellon University, Pittsburgh, PA, 2024. [Online]. Available: https://www.ml.cmu.edu/research/cchallu_phd_mld_2024.pdf

[13] S. Park, S. Han, and S. S. Woo, "Forecasting Error Pattern-Based Anomaly Detection in Multivariate Time Series," in *Machine Learning and Knowledge Discovery in Databases: Applied Data Science Track: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part IV*, Berlin, Heidelberg: Springer-Verlag, Sep. 2020, pp. 157–172. doi: 10.1007/978-3-030-67667-4_10.

[14] X. Hu and L. Liu, "An Attack Prediction and Recognition Method for Water Treatment System Based on One-dimensional Convolutional Neural Network and Cumulative Sum," in *2024 4th International Conference on Neural Networks, Information and Communication Engineering (NNICE)*, Jan. 2024, pp. 715–718. doi: 10.1109/NNICE61279.2024.10498833.

[15] J. Goh, S. Adepu, K. N. Junejo, and A. Mathur, "A Dataset to Support Research in the Design of Secure Water Treatment Systems," in *Critical Information Infrastructures Security*, G. Havarneanu, R. Setola, H. Nassopoulos, and S. Wolthusen, Eds., Cham: Springer International Publishing, 2017, pp. 88–99. doi: 10.1007/978-3-319-71368-7_8.

[16] N. Muralidhar, S. Muthiah, K. Nakayama, R. Sharma, and N. Ramakrishnan, "Multivariate Long-Term State Forecasting in Cyber-Physical Systems: A Sequence to Sequence Approach," in *2019 IEEE International Conference on Big Data (Big Data)*, Dec. 2019, pp. 543–552. doi: 10.1109/BigData47090.2019.9005511.

[17] K. Team, "Keras documentation: ModelCheckpoint." Accessed: Jul. 18, 2024. [Online]. Available: https://keras.io/api/callbacks/model_checkpoint/

[18] K. Team, "Keras documentation: EarlyStopping." Accessed: Jul. 18, 2024. [Online]. Available: https://keras.io/api/callbacks/early_stopping/

[19] K. Team, "Keras documentation: ReduceLROnPlateau." Accessed: Jul. 18, 2024. [Online]. Available: https://keras.io/api/callbacks/reduce_lr_on_plateau/

[20] N. Mehdiyev, D. Enke, P. Fettke, and P. Loos, "Evaluating Forecasting Methods by Considering Different Accuracy Measures," *Procedia Comput. Sci.*, vol. 95, pp. 264–271, Jan. 2016, doi: 10.1016/j.procs.2016.09.332.

[21] A. Botchkarev, "A New Typology Design of Performance Metrics to Measure Errors in Machine Learning Regression Algorithms," *Interdiscip. J. Inf. Knowl. Manag.*, vol. 14, pp. 045–076, 2019, doi: 10.28945/4184.

[22] C.-L. Cheng, Shalabh, and G. Garg, "Coefficient of determination for multiple measurement error models," *J. Multivar. Anal.*, vol. 126, pp. 137–152, Apr. 2014, doi: 10.1016/j.jmva.2014.01.006.

[23] W. Bischoff, "Loss Function," in *International Encyclopedia of Statistical Science*, M. Lovric, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 762–763. doi: 10.1007/978-3-642-04898-2_621.

[24] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," arXiv.org. Accessed: Jul. 16, 2024. [Online]. Available: https://arxiv.org/abs/1412.6980v9

[25] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, "Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark," arXiv.org. Accessed: Jul. 16, 2024. [Online]. Available: https://arxiv.org/abs/2109.14545v3

[26] S. Amini, A. Azizian, and P. Daneshkar Arasteh, "Improving the Performance of Global Rainfall Forecasting Systems in Different Climate Areas of Iran Using Quantile Mapping Method," *Iran. J. Soil Water Res.*, vol. 51, no. 9, pp. 2275–2291, Nov. 2020, doi: 10.22059/ijswr.2020.302208.668602.

[27] S. Jagadeesh, "Multivariate Multi-step Time Series Forecasting using Stacked LSTM sequence to sequence Autoencoder in Tensorflow 2.0 / Keras," Analytics Vidhya. Accessed: Jul. 18, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/10/multivariate-multi-step-time-series-forecasting-using-stacked-lstm-sequence-to-sequence-autoencoder-in-tensorflow-2-0-keras/

[28] T. Theodoropoulos, A.-C. Maroudis, J. Violos, and K. Tserpes, "An Encoder-Decoder Deep Learning Approach for Multistep Service Traffic Prediction," in *2021 IEEE Seventh International Conference on Big Data Computing Service and Applications (BigDataService)*, Aug. 2021, pp. 33–40. doi: 10.1109/BigDataService52369.2021.00010.

[29] Y. Huang and Y. Sun, "Learning to Pour," *arXiv*, vol. 1705.09021, pp. 1–7, May 2017.

[30] A. I. Shahin and S. Almotairi, "A Deep Learning BiLSTM Encoding-Decoding Model for COVID-19 Pandemic Spread Forecasting," *Fractal Fract.*, vol. 5, no. 4, Art. no. 4, Dec. 2021, doi: 10.3390/fractalfract5040175.

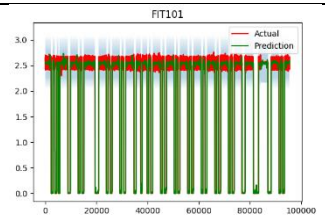
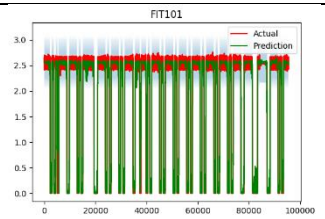
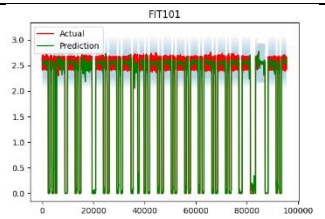
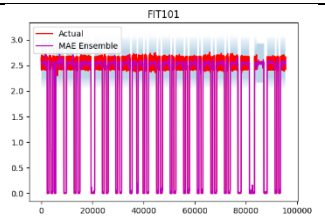
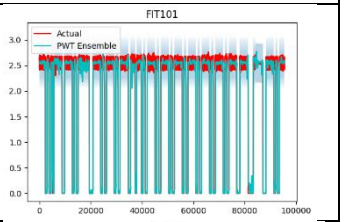
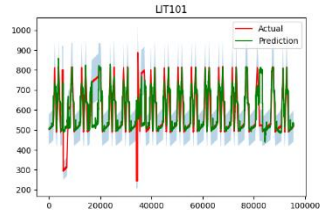
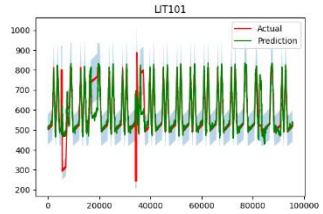
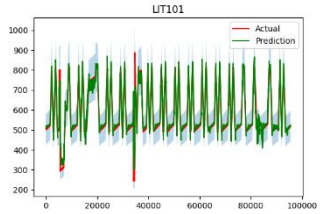
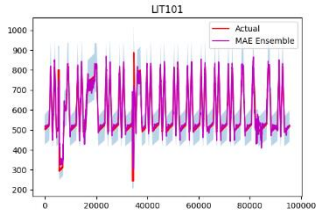
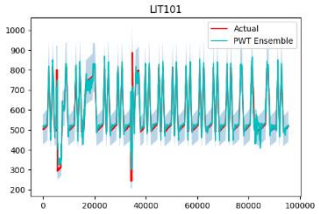
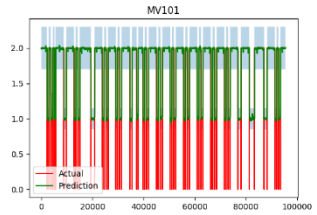
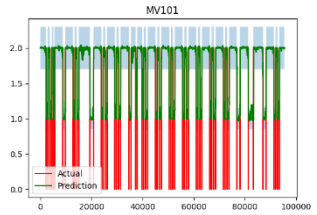
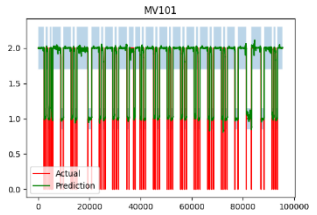
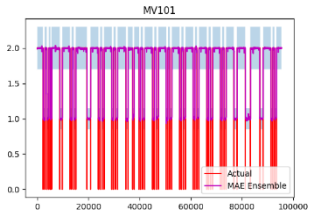
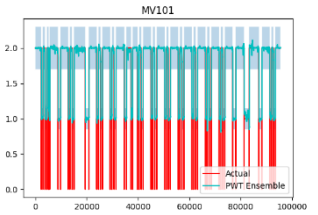
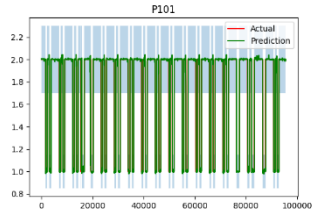
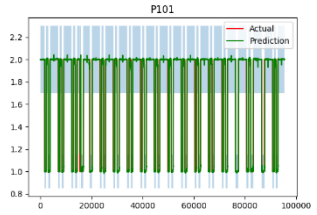
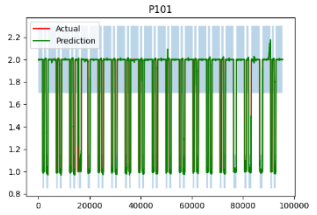
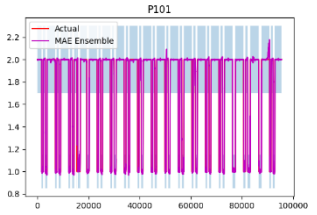
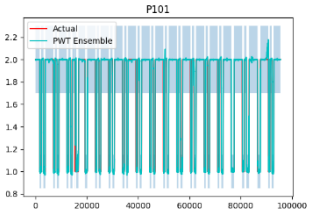
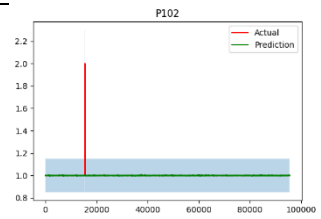
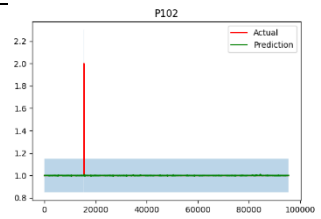
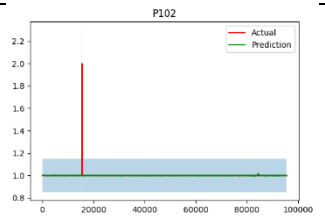
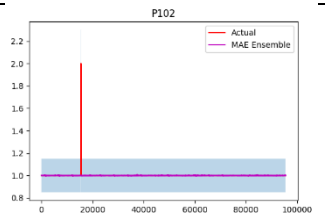
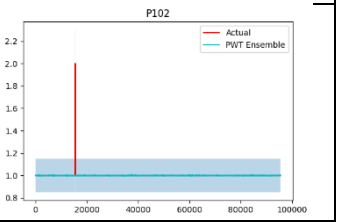
[31] M. Phi, "Illustrated Guide to LSTM's and GRU's: A step by step explanation," Medium. Accessed: Jul. 18, 2024. [Online]. Available: https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

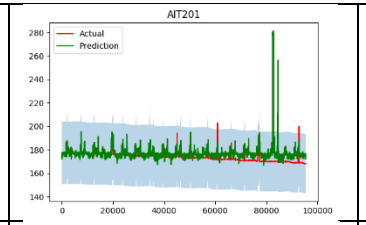
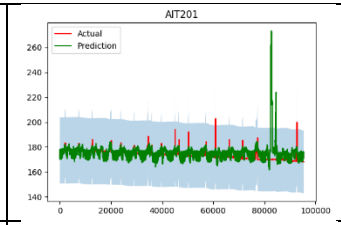
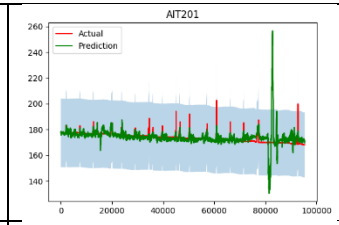
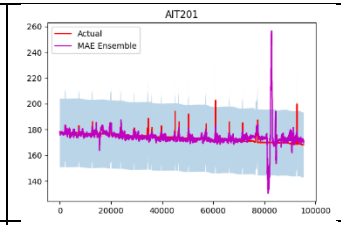
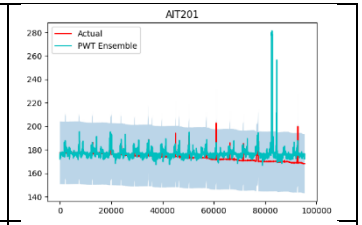
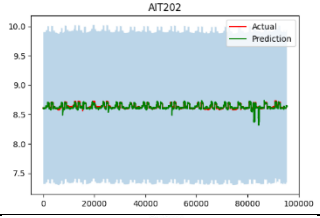
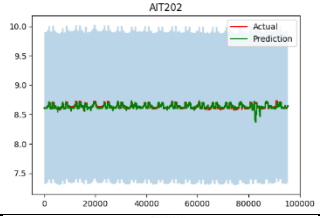
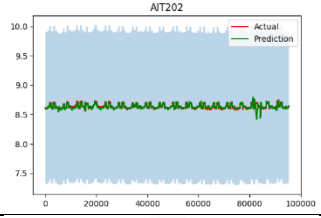
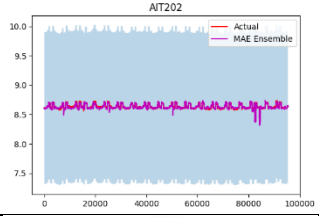
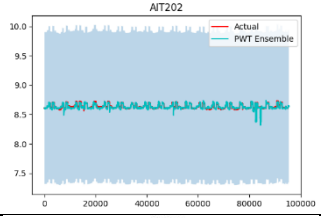
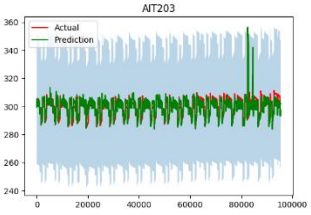
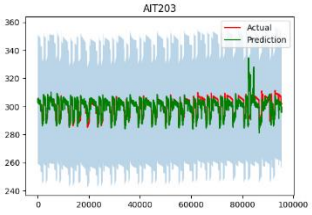
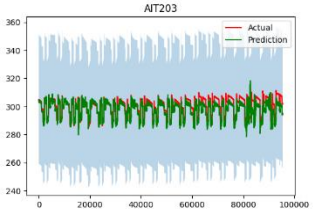
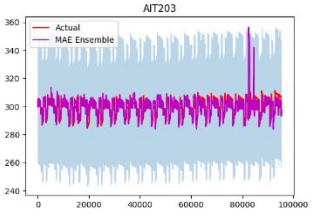
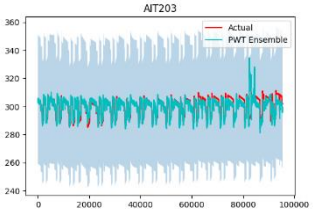
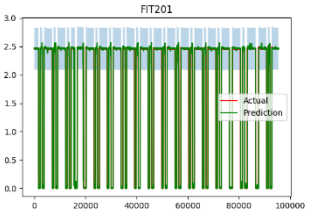
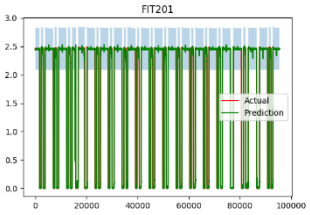
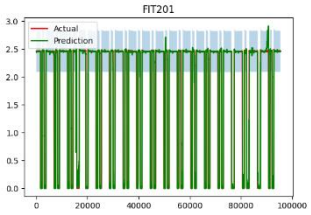
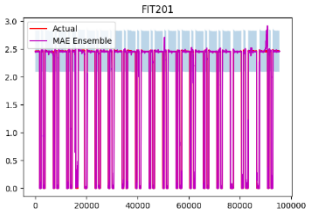
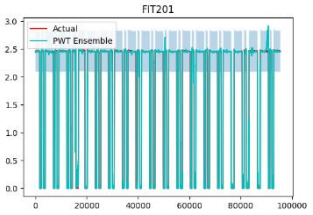
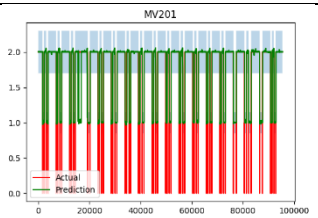
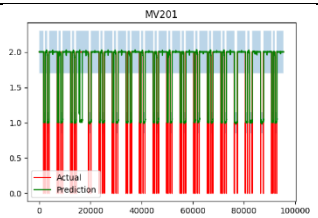
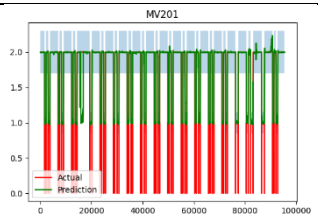
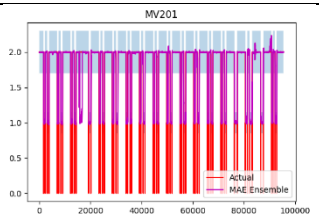
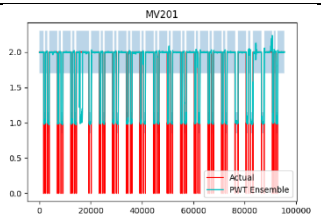
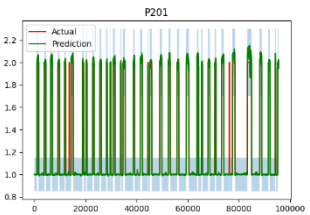
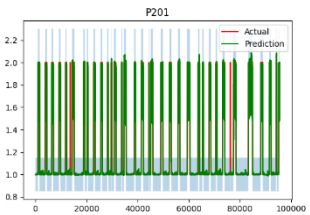
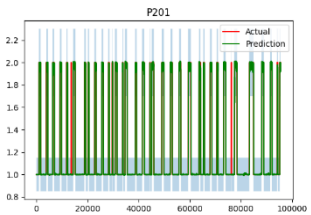
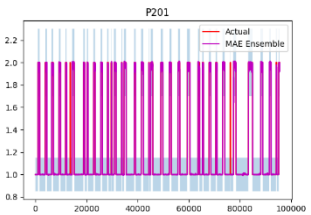
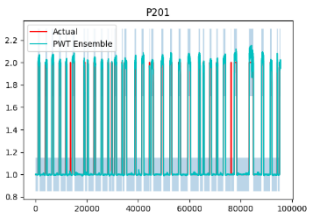
[32] S. Nosouhian, F. Nosouhian, and A. K. Khoshouei, "A Review of Recurrent Neural Network Architecture for Sequence Learning: Comparison between LSTM and GRU," Jul. 12, 2021, *Preprints*: 2021070252. doi: 10.20944/preprints202107.0252.v1.

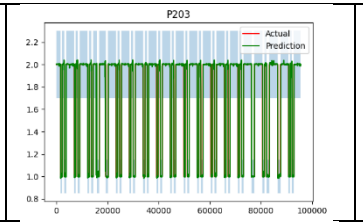
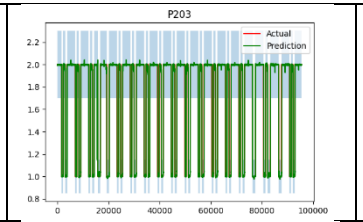
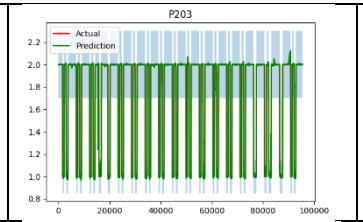
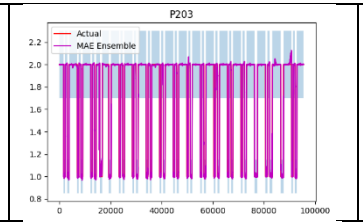
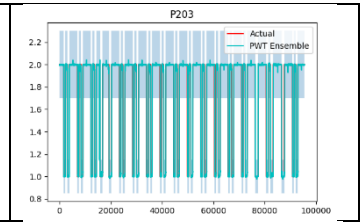
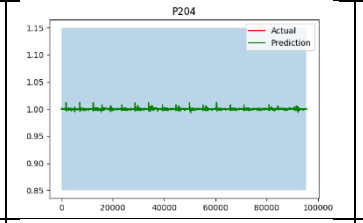
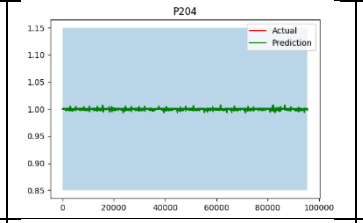
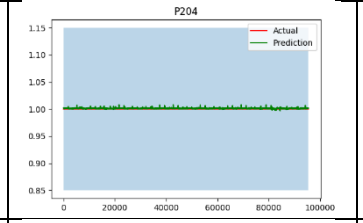
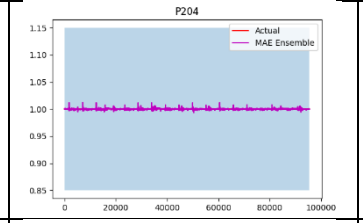
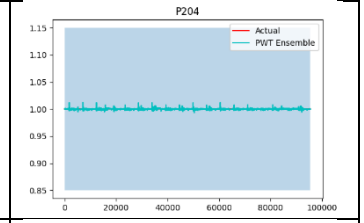
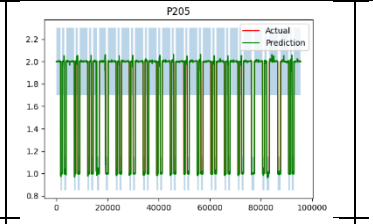
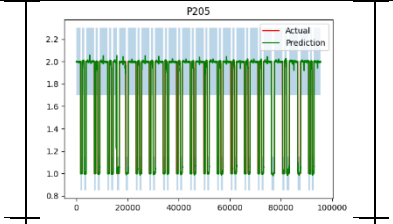
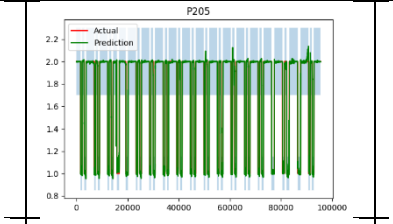
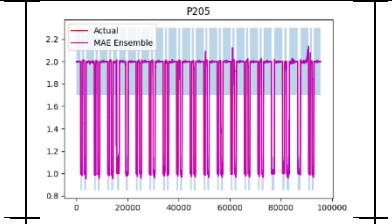
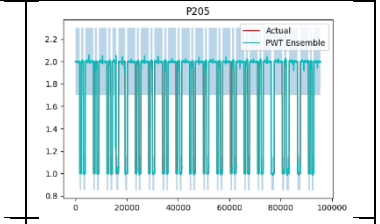
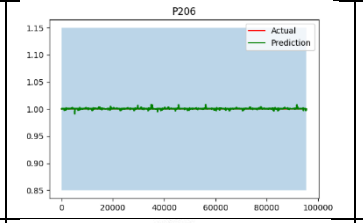
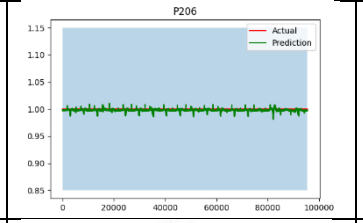
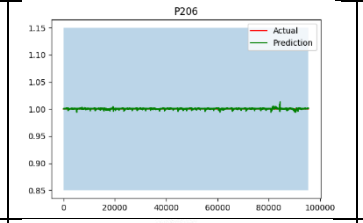
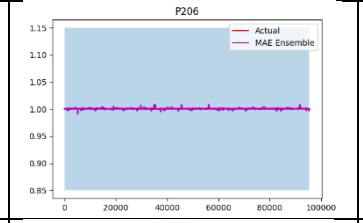
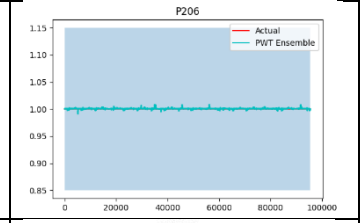
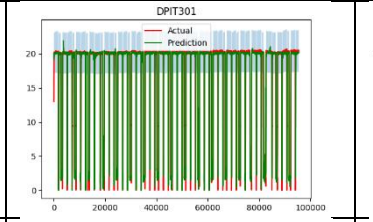
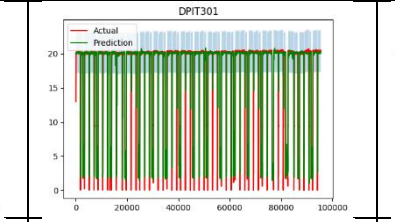
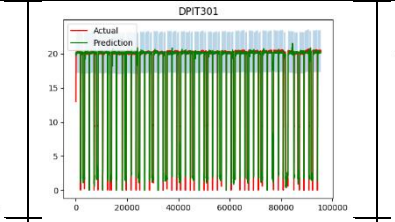
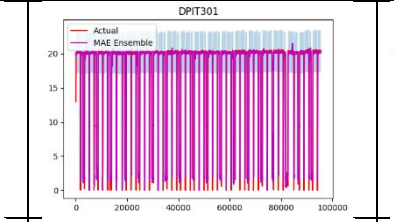
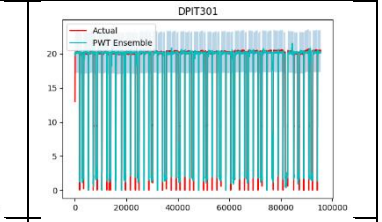
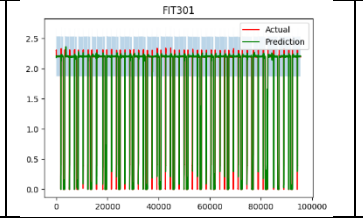
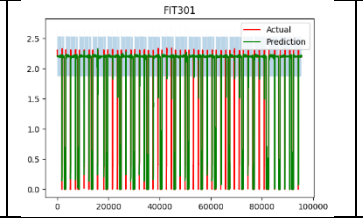
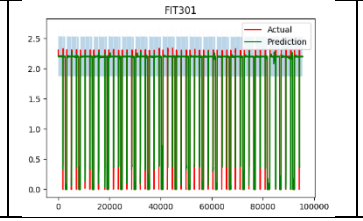
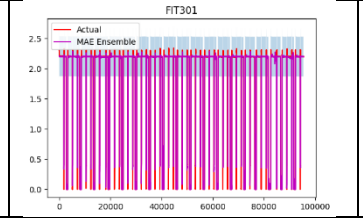
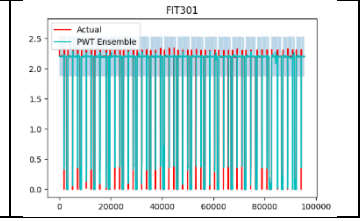
[33] H. Balti, A. Ben Abbes, and I. R. Farah, "A Bi-GRU-based encoder–decoder framework for multivariate time series forecasting," *Soft Comput.*, vol. 28, no. 9–10, pp. 6775–6786, May 2024, doi: 10.1007/s00500-023-09531-9.

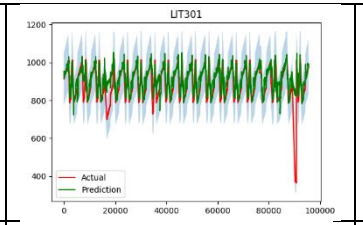
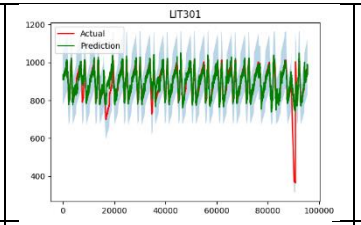
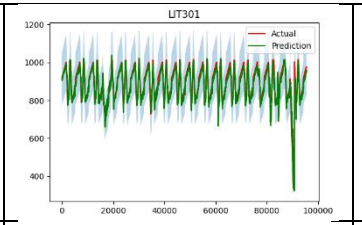
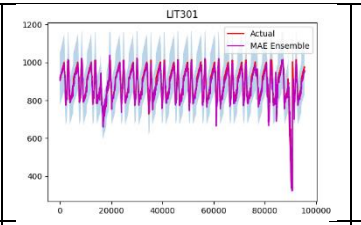
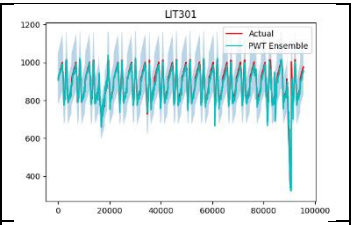
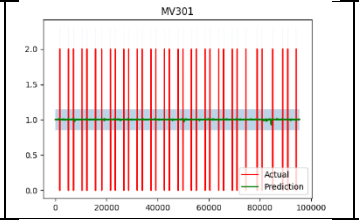
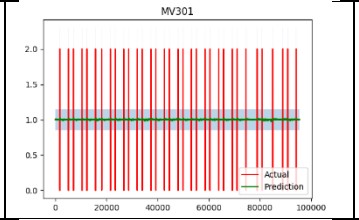
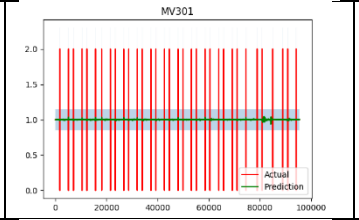
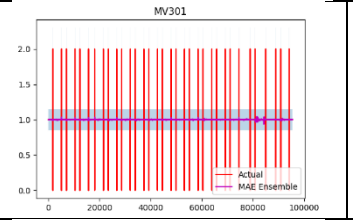
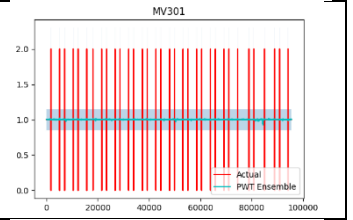
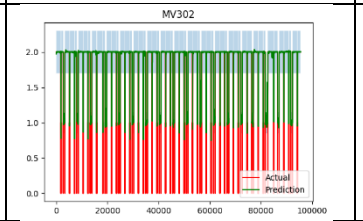
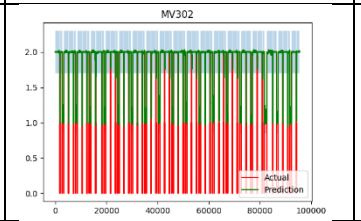
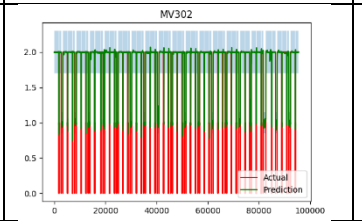
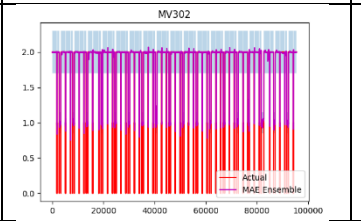
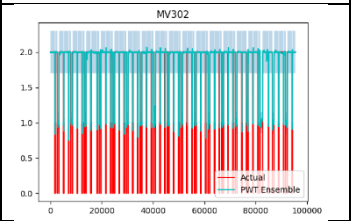
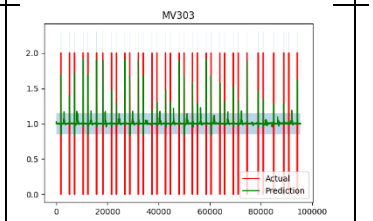
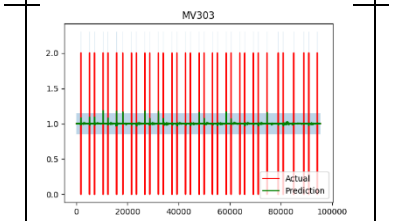
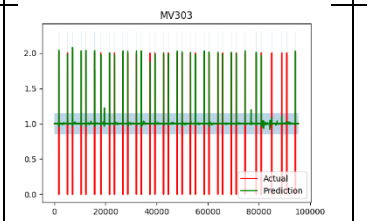
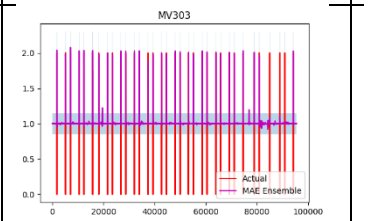
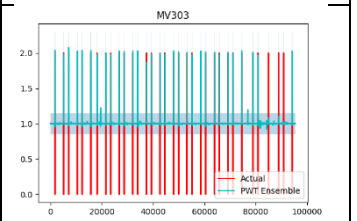
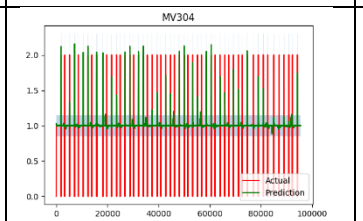
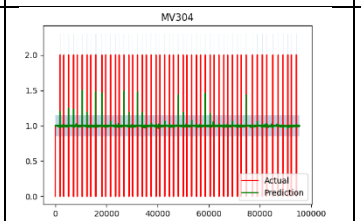
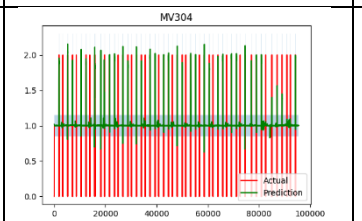
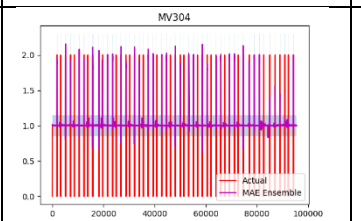
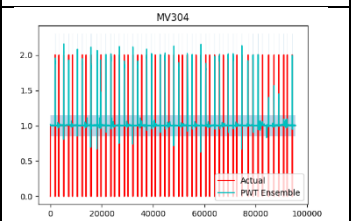
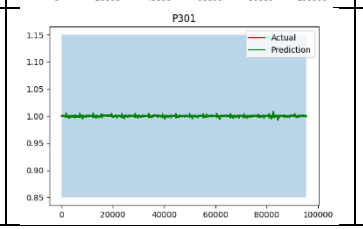
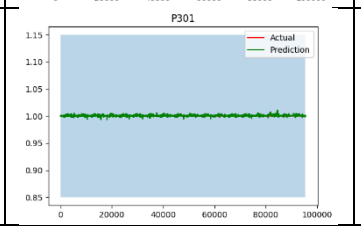
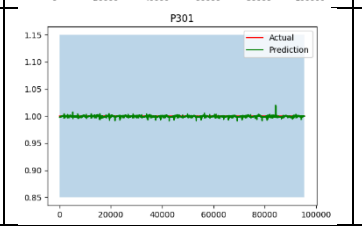
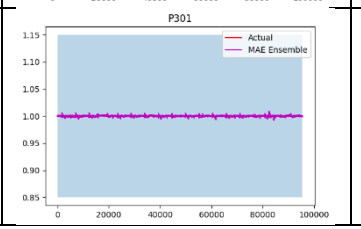
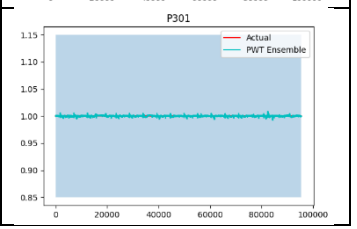
## VII. APPENDIX

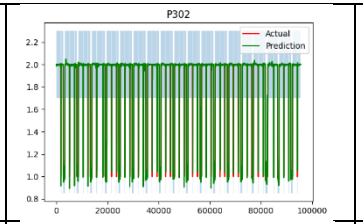
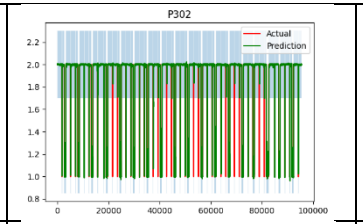
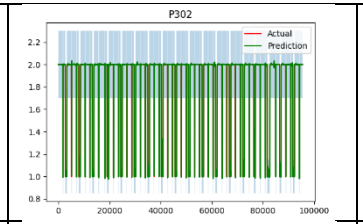
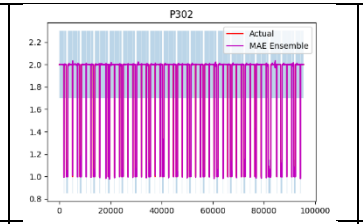
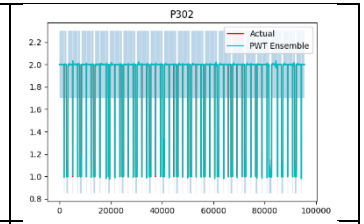
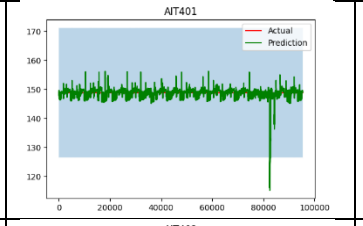
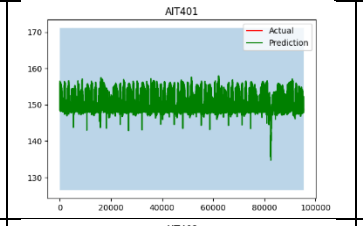
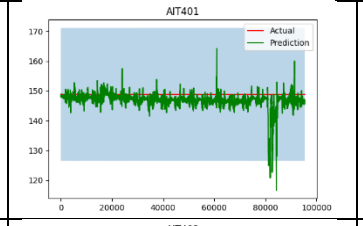
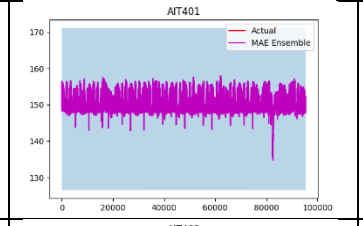
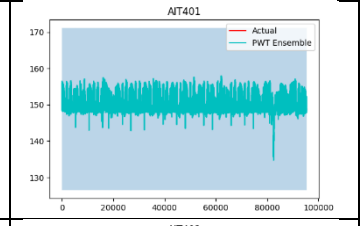
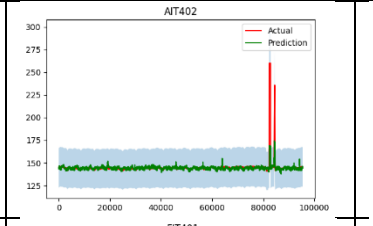
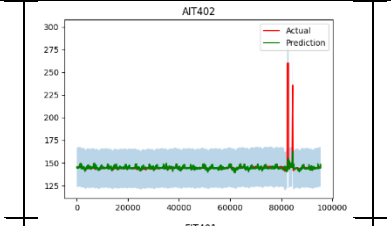
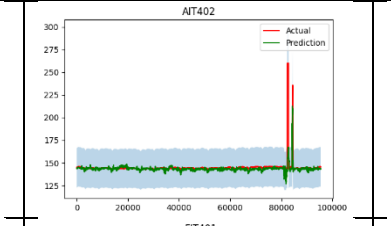
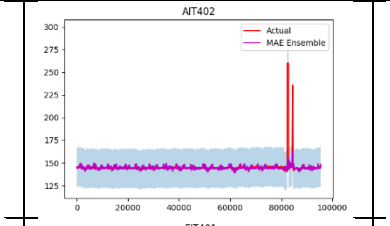
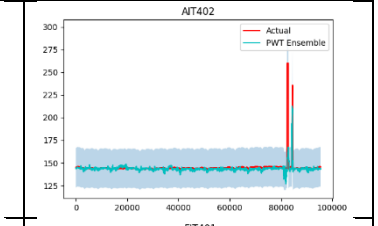
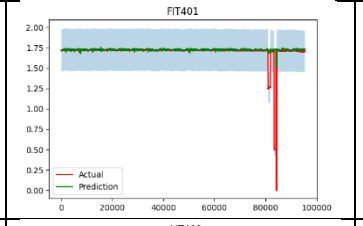
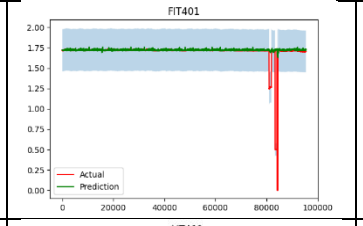
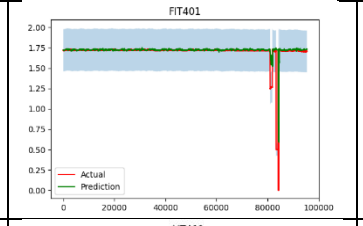
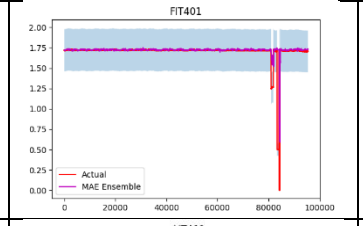
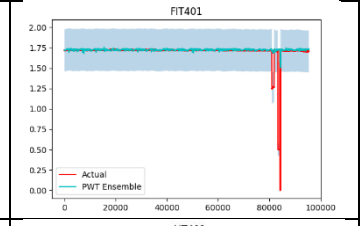
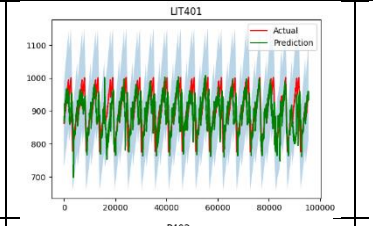
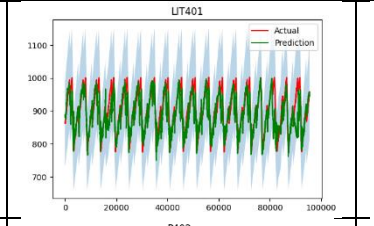
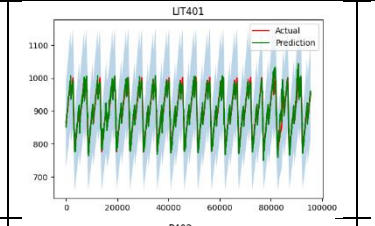
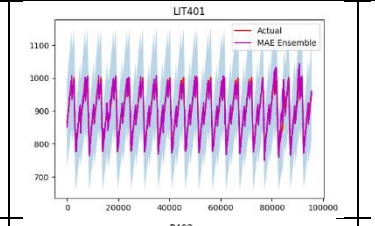
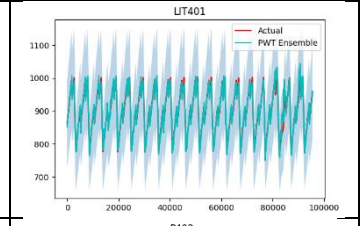
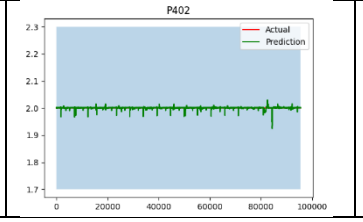
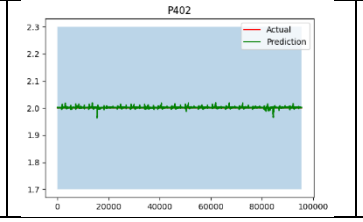
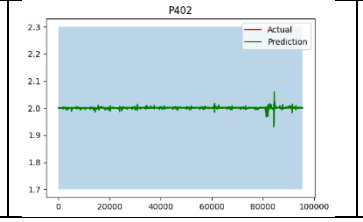
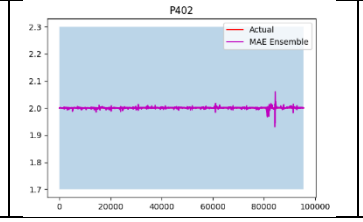
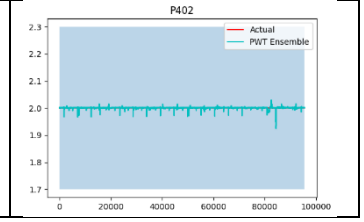
### *A. Models' Actual vs. Predicted Plots for Different Forecasting Models*

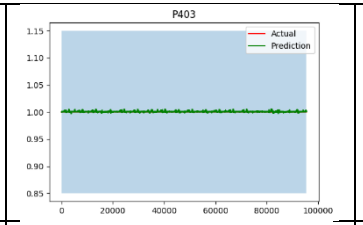
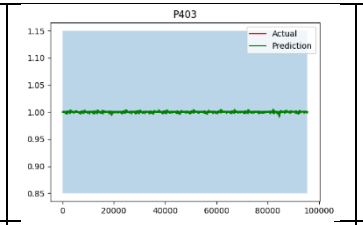
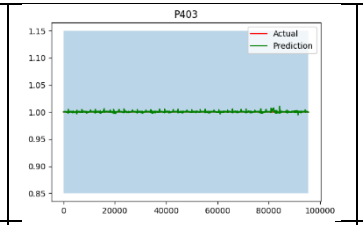
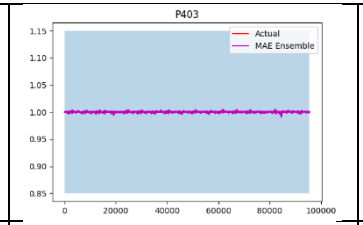
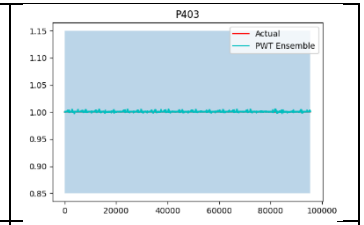
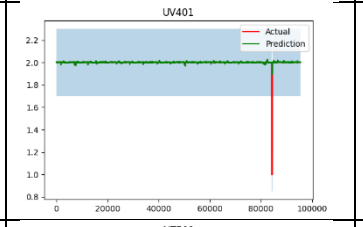
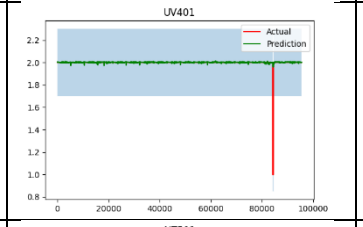
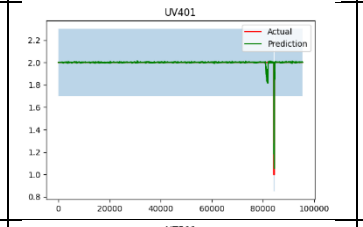
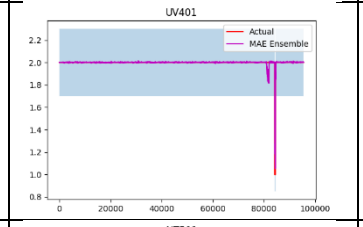
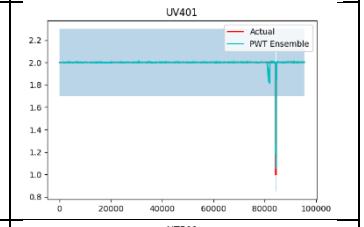
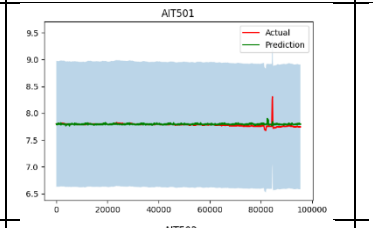
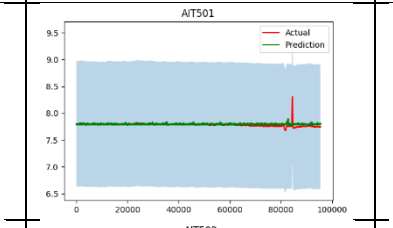
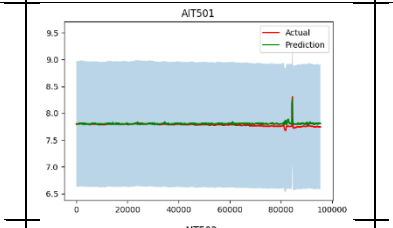
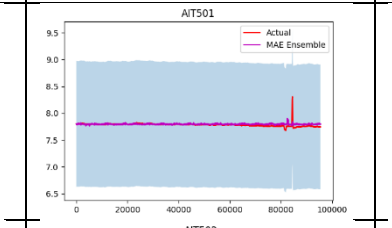
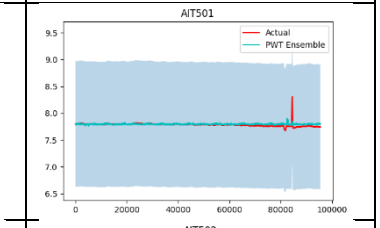
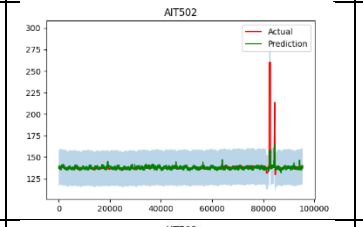
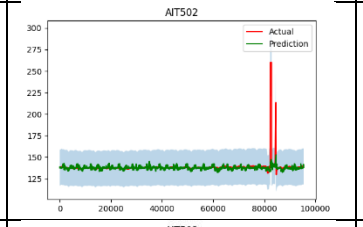
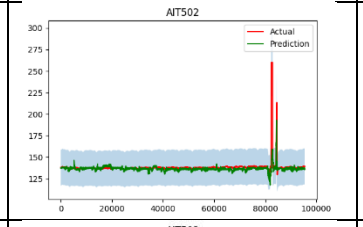
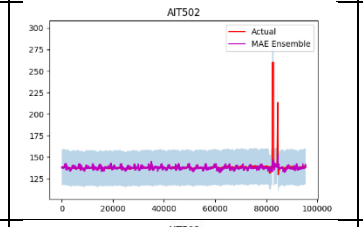
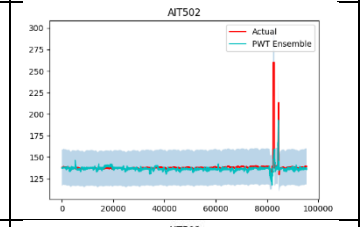
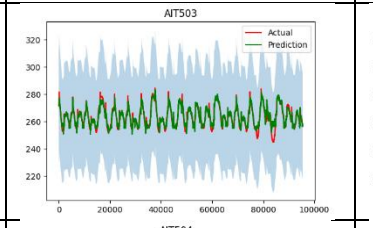
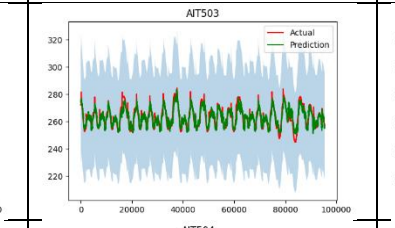
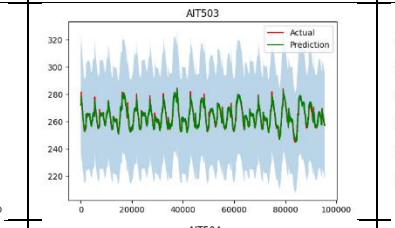
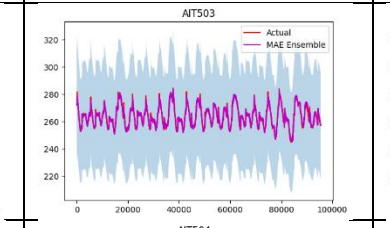
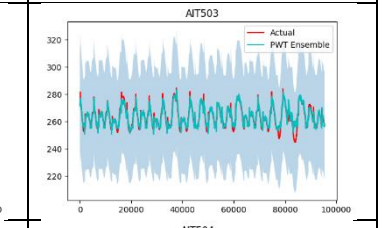
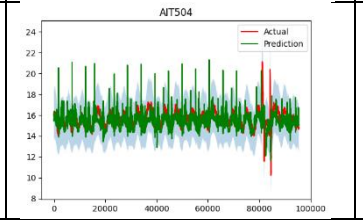
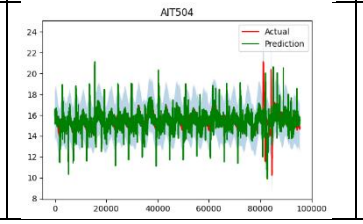
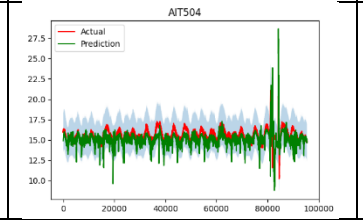
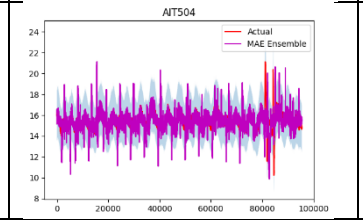
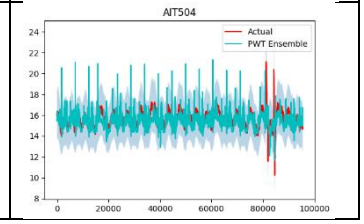
| Feature | LSTM | BiLSTM | BiGRU | Min MAE Ensemble | Max PWT Ensemble |
|---|---|---|---|---|---|
| FIT101 |  |  |  |  |  |
| LIT101 |  |  |  |  |  |
| MV101 |  |  |  |  |  |
| P101 |  |  |  |  |  |
| P102 |  |  |  |  |  |

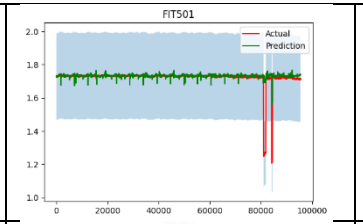
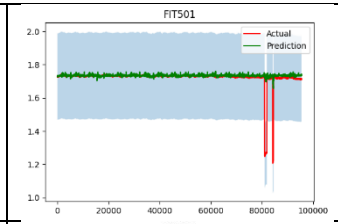
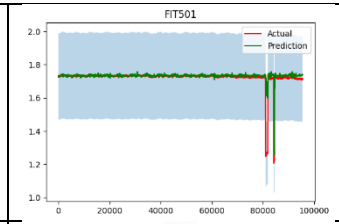
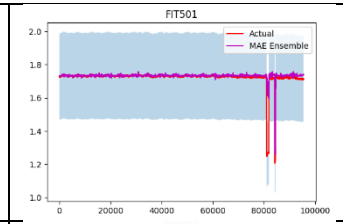
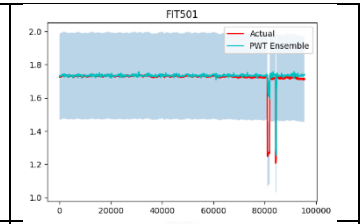
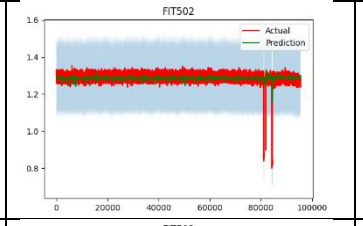
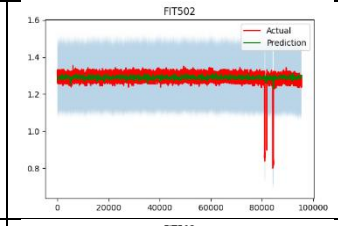
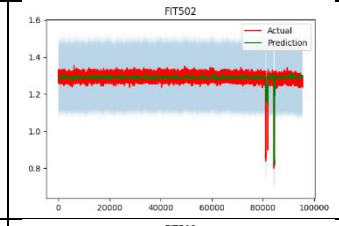
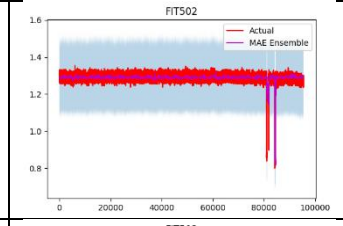
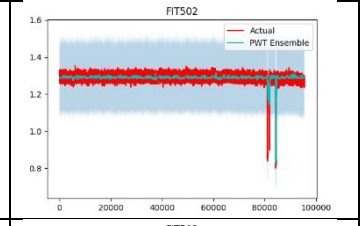
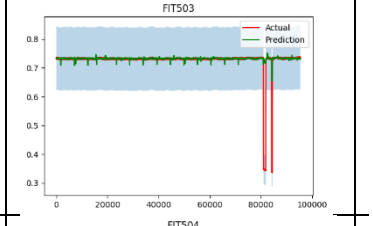
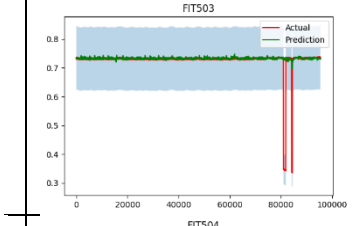
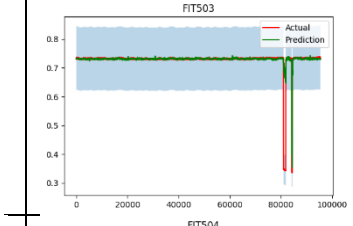
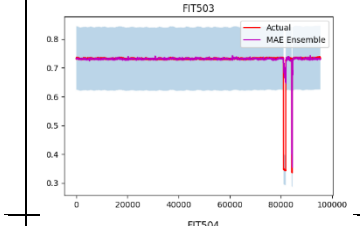
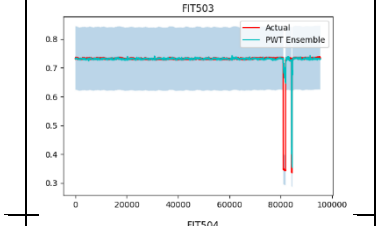
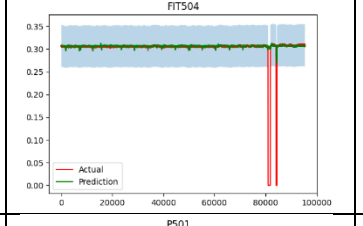
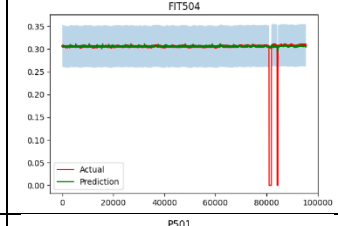
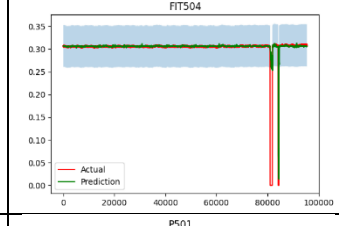
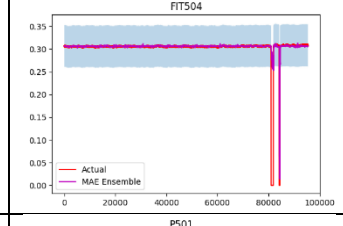
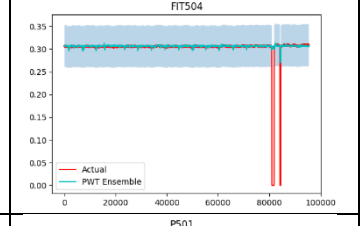
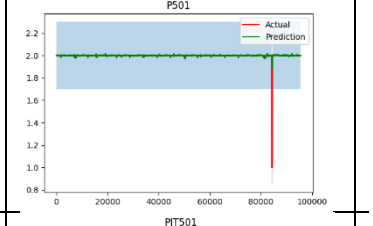
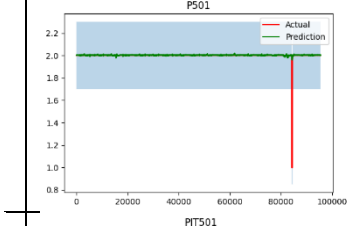
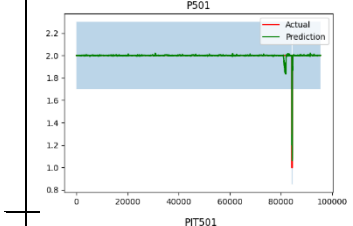
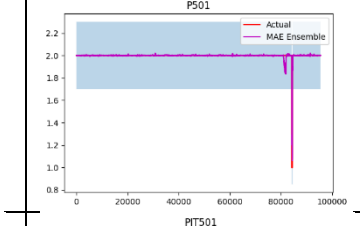
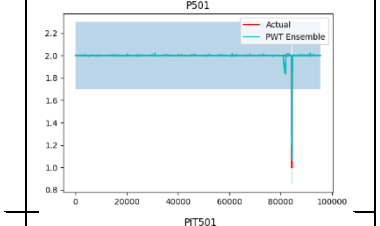
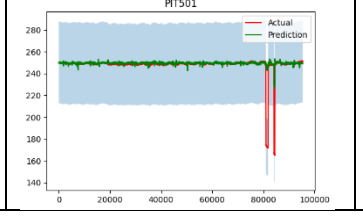
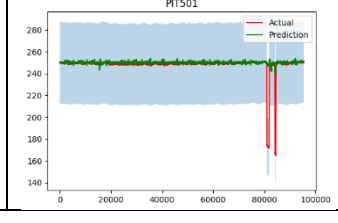
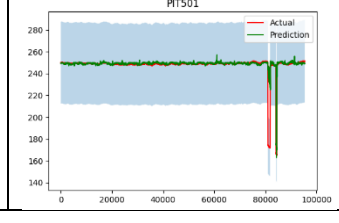
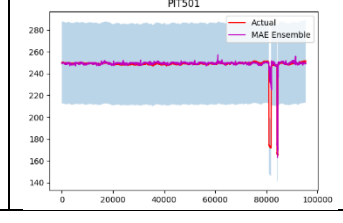
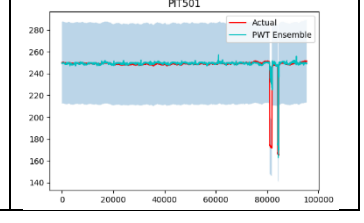
| | | | | | |
|---|---|---|---|---|---|
| AIT201 |  |  |  |  |  |
| AIT202 |  |  |  |  |  |
| AIT203 |  |  |  |  |  |
| FIT201 |  |  |  |  |  |
| MV201 |  |  |  |  |  |
| P201 |  |  |  |  |  |

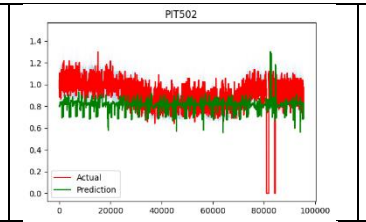
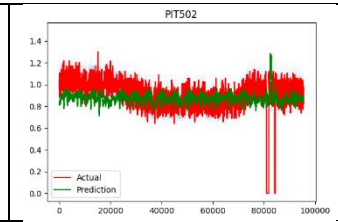
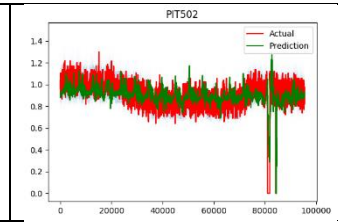
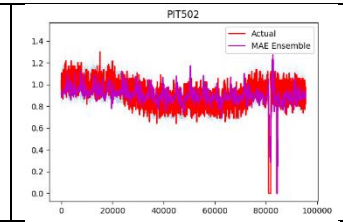
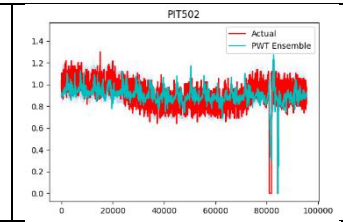
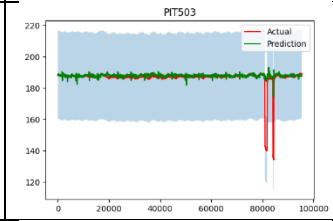
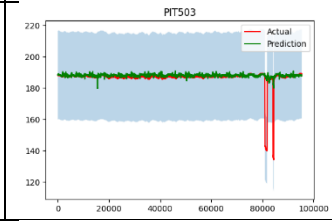
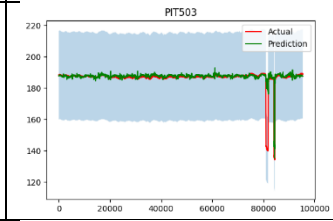
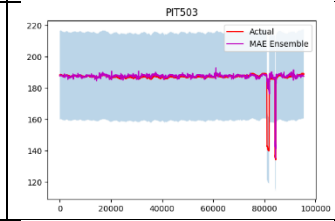
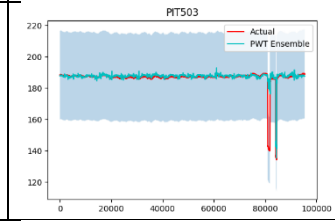
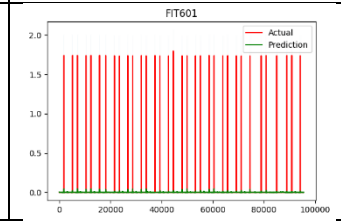
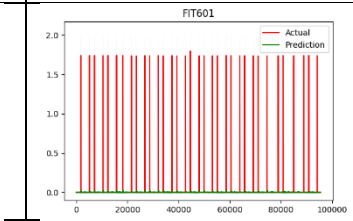
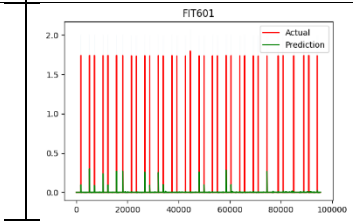
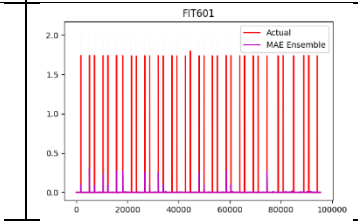
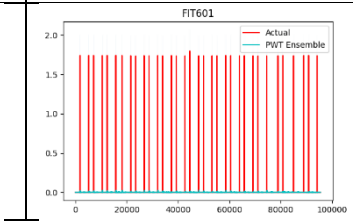
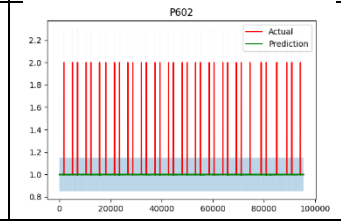
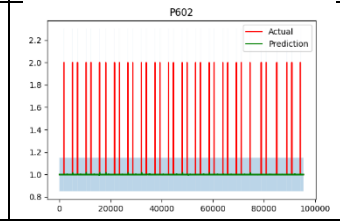
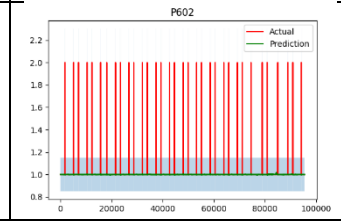
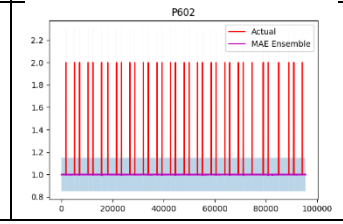
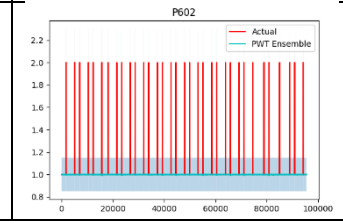
| | | | | | |
|---|---|---|---|---|---|
| P203 |  |  |  |  |  |
| P204 |  |  |  |  |  |
| P205 |  |  |  |  |  |
| P206 |  |  |  |  |  |
| DPIT301 |  |  |  |  |  |
| FIT301 |  |  |  |  |  |

| | | | | | |
|---|---|---|---|---|---|
| LIT301 |  |  |  |  |  |
| MV301 |  |  |  |  |  |
| MV302 |  |  |  |  |  |
| MV303 |  |  |  |  |  |
| MV304 |  |  |  |  |  |
| P301 |  |  |  |  |  |

| | | | | | |
|---|---|---|---|---|---|
| P302 |  |  |  |  |  |
| AIT401 |  |  |  |  |  |
| AIT402 |  |  |  |  |  |
| FIT401 |  |  |  |  |  |
| LIT401 |  |  |  |  |  |
| P402 |  |  |  |  |  |

| | | | | | |
|---|---|---|---|---|---|
| P403 |  |  |  |  |  |
| UV401 |  |  |  |  |  |
| AIT501 |  |  |  |  |  |
| AIT502 |  |  |  |  |  |
| AIT503 |  |  |  |  |  |
| AIT504 |  |  |  |  |  |

| | | | | | |
|---|---|---|---|---|---|
| FIT501 |  |  |  |  |  |
| FIT502 |  |  |  |  |  |
| FIT503 |  |  |  |  |  |
| FIT504 |  |  |  |  |  |
| P501 |  |  |  |  |  |
| PIT501 |  |  |  |  |  |

| | | | | | |
|---|---|---|---|---|---|
| PIT502 |  |  |  |  |  |
| PIT503 |  |  |  |  |  |
| FIT601 |  |  |  |  |  |
| P602 |  |  |  |  |  |

*B. PWT Before and After Quantile Mapping Comparison*

*1) LSTM Model*

| Feature | PWT Before Quantile Mapping (%) | PWT After Quantile Mapping (%) |
|---|---|---|
| FIT101 | 81.06 | 94.27 |
| LIT101 | 82.59 | 77.95 |
| MV101 | 95.22 | 93.20 |
| P101 | 95.68 | 94.58 |
| P102 | 99.90 | 99.50 |
| AIT201 | 99.25 | 25.40 |
| AIT202 | 100.00 | 99.12 |
| AIT203 | 99.50 | 90.34 |
| FIT201 | 77.75 | 94.25 |
| MV201 | 94.45 | 92.55 |
| P201 | 95.98 | 82.52 |
| P203 | 96.11 | 94.82 |
| P204 | 100.00 | 99.94 |
| P205 | 96.12 | 90.34 |
| P206 | 100.00 | 99.82 |
| DPIT301 | 88.78 | 83.47 |
| FIT301 | 86.03 | 88.16 |
| LIT301 | 96.05 | 94.01 |
| MV301 | 98.30 | 97.95 |
| MV302 | 94.35 | 89.29 |
| MV303 | 96.18 | 96.63 |
| MV304 | 95.70 | 93.99 |
| P301 | 100.00 | 99.96 |
| P302 | 94.95 | 89.42 |
| AIT401 | 99.75 | 91.16 |
| AIT402 | 99.16 | 70.44 |
| FIT401 | 97.95 | 94.73 |
| LIT401 | 98.53 | 90.56 |
| P402 | 100.00 | 96.60 |
| P403 | 100.00 | 99.47 |
| UV401 | 99.69 | 96.71 |
| AIT501 | 100.00 | 100.00 |
| AIT502 | 99.22 | 72.95 |
| AIT503 | 100.00 | 100.00 |
| AIT504 | 98.75 | 36.96 |
| FIT501 | 98.76 | 95.66 |
| FIT502 | 99.50 | 95.81 |
| FIT503 | 98.75 | 96.18 |
| FIT504 | 98.75 | 97.01 |
| P501 | 99.69 | 96.26 |
| PIT501 | 98.76 | 96.03 |
| PIT502 | 69.93 | 34.06 |
| PIT503 | 98.76 | 95.84 |
| FIT601 | 76.21 | 0.29 |
| P602 | 98.90 | 98.63 |

*2) BiLSTM Model*

| Feature | PWT Before Quantile Mapping (%) | PWT After Quantile Mapping (%) |
|---|---|---|
| FIT101 | 73.45 | 94.81 |
| LIT101 | 90.90 | 89.70 |
| MV101 | 94.36 | 93.38 |
| P101 | 95.69 | 95.02 |
| P102 | 99.90 | 99.36 |
| AIT201 | 98.99 | 24.80 |
| AIT202 | 100.00 | 99.37 |
| AIT203 | 100.00 | 90.03 |
| FIT201 | 77.17 | 94.72 |
| MV201 | 94.49 | 93.48 |
| P201 | 95.82 | 82.00 |
| P203 | 96.15 | 95.63 |
| P204 | 100.00 | 99.48 |
| P205 | 96.15 | 90.63 |
| P206 | 100.00 | 99.28 |
| DPIT301 | 87.96 | 82.40 |
| FIT301 | 84.92 | 87.70 |
| LIT301 | 96.36 | 95.66 |
| MV301 | 98.30 | 97.74 |
| MV302 | 94.89 | 90.35 |
| MV303 | 95.83 | 95.92 |
| MV304 | 94.65 | 93.60 |
| P301 | 100.00 | 99.79 |
| P302 | 94.64 | 89.51 |
| AIT401 | 100.00 | 91.12 |
| AIT402 | 99.22 | 70.28 |
| FIT401 | 97.95 | 94.70 |
| LIT401 | 99.42 | 92.87 |
| P402 | 100.00 | 96.40 |
| P403 | 100.00 | 99.50 |
| UV401 | 99.69 | 96.36 |
| AIT501 | 100.00 | 100.00 |
| AIT502 | 99.20 | 73.50 |
| AIT503 | 100.00 | 100.00 |
| AIT504 | 98.10 | 36.90 |
| FIT501 | 98.76 | 95.93 |
| FIT502 | 99.50 | 95.85 |
| FIT503 | 98.75 | 95.92 |
| FIT504 | 98.74 | 96.58 |
| P501 | 99.69 | 96.51 |
| PIT501 | 98.76 | 95.83 |
| PIT502 | 87.94 | 34.07 |
| PIT503 | 98.76 | 95.90 |
| FIT601 | 76.73 | 0.32 |
| P602 | 98.90 | 98.83 |

*3) BiGRU Model*

| Feature | PWT Before Quantile Mapping (%) | PWT After Quantile Mapping (%) |
|---|---|---|
| FIT101 | 82.72 | 95.11 |
| LIT101 | 98.33 | 98.39 |
| MV101 | 95.23 | 94.23 |
| P101 | 95.78 | 95.02 |
| P102 | 99.90 | 99.53 |
| AIT201 | 98.58 | 25.14 |
| AIT202 | 100.00 | 99.62 |
| AIT203 | 100.00 | 90.16 |
| FIT201 | 84.01 | 94.86 |
| MV201 | 94.76 | 93.28 |
| P201 | 95.98 | 82.24 |
| P203 | 95.63 | 95.46 |
| P204 | 100.00 | 99.23 |
| P205 | 96.00 | 90.43 |
| P206 | 100.00 | 99.62 |
| DPIT301 | 90.61 | 83.70 |
| FIT301 | 89.46 | 87.64 |
| LIT301 | 99.36 | 98.81 |
| MV301 | 98.30 | 97.61 |
| MV302 | 95.52 | 89.90 |
| MV303 | 97.52 | 97.09 |
| MV304 | 96.44 | 94.38 |
| P301 | 100.00 | 99.73 |
| P302 | 95.27 | 89.70 |
| AIT401 | 99.73 | 91.03 |
| AIT402 | 99.27 | 70.34 |
| FIT401 | 97.95 | 94.85 |
| LIT401 | 100.00 | 93.99 |
| P402 | 100.00 | 96.28 |
| P403 | 100.00 | 99.51 |
| UV401 | 99.79 | 96.42 |
| AIT501 | 100.00 | 100.00 |
| AIT502 | 99.31 | 73.07 |
| AIT503 | 100.00 | 100.00 |
| AIT504 | 97.27 | 36.01 |
| FIT501 | 99.00 | 96.03 |
| FIT502 | 99.74 | 95.97 |
| FIT503 | 98.87 | 96.15 |
| FIT504 | 98.73 | 97.19 |
| P501 | 99.79 | 96.46 |
| PIT501 | 99.00 | 96.03 |
| PIT502 | 90.49 | 35.25 |
| PIT503 | 99.00 | 96.09 |
| FIT601 | 48.84 | 2.16 |
| P602 | 98.90 | 98.54 |

*C. Model Selection Per Feature for Ensemble Models*

*1) Ensemble Model 1: Lowest Mae for Each Feature*

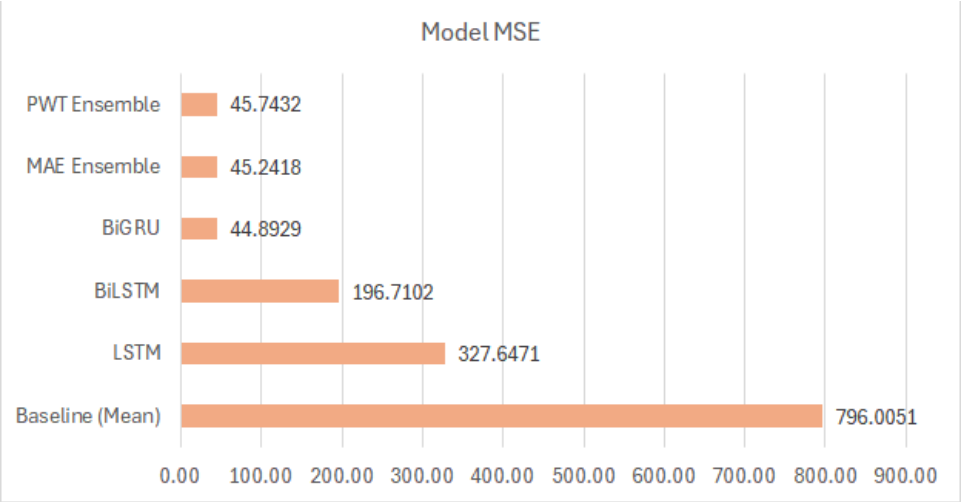
| Feature | LSTM MAE | BiLSTM MAE | BiGRU MAE | Selected Model |
|---|---|---|---|---|
| FIT101 | 0.15057 | 0.162903 | 0.152645 | LSTM |
| LIT101 | 48.784025 | 31.006983 | 12.800767 | BiGRU |
| MV101 | 0.053704 | 0.061278 | 0.058321 | LSTM |
| P101 | 0.044601 | 0.044027 | 0.040448 | BiGRU |
| P102 | 0.001354 | 0.001866 | 0.001544 | LSTM |
| AIT201 | 4.119347 | 3.697466 | 2.642209 | BiGRU |
| AIT202 | 0.014209 | 0.015732 | 0.015186 | LSTM |
| AIT203 | 2.630104 | 2.706319 | 3.306439 | LSTM |
| FIT201 | 0.099436 | 0.096974 | 0.086431 | BiGRU |
| MV201 | 0.061848 | 0.060797 | 0.055694 | BiGRU |
| P201 | 0.04447 | 0.04598 | 0.042316 | BiGRU |
| P203 | 0.040103 | 0.039601 | 0.038011 | BiGRU |
| P204 | 0.000517 | 0.000595 | 0.001485 | LSTM |
| P205 | 0.040605 | 0.039958 | 0.037906 | BiGRU |
| P206 | 0.000723 | 0.001805 | 0.001118 | LSTM |
| DPIT301 | 0.954189 | 0.999285 | 0.894004 | BiGRU |
| FIT301 | 0.096283 | 0.107546 | 0.092045 | BiGRU |
| LIT301 | 37.237777 | 26.228298 | 15.778221 | BiGRU |
| MV301 | 0.01976 | 0.020297 | 0.018526 | BiGRU |
| MV302 | 0.060359 | 0.060625 | 0.053133 | BiGRU |
| MV303 | 0.036196 | 0.041629 | 0.028686 | BiGRU |
| MV304 | 0.039325 | 0.048254 | 0.035446 | BiGRU |
| P301 | 0.000471 | 0.000692 | 0.001018 | LSTM |
| P302 | 0.04835 | 0.054967 | 0.048155 | BiGRU |
| AIT401 | 0.654162 | 0.534723 | 1.939546 | BiLSTM |
| AIT402 | 1.598922 | 1.543094 | 2.16114 | BiLSTM |
| FIT401 | 0.023846 | 0.025589 | 0.021896 | BiGRU |
| LIT401 | 36.044982 | 24.264048 | 6.764726 | BiGRU |
| P402 | 0.001566 | 0.002588 | 0.001367 | BiGRU |
| P403 | 0.000918 | 0.00061 | 0.000646 | BiLSTM |
| UV401 | 0.003956 | 0.004387 | 0.003419 | BiGRU |
| AIT501 | 0.018239 | 0.02314 | 0.024316 | LSTM |
| AIT502 | 1.685986 | 1.671138 | 2.615002 | BiLSTM |
| AIT503 | 1.617058 | 2.602074 | 0.833081 | BiGRU |
| AIT504 | 0.518845 | 0.404398 | 0.698105 | BiLSTM |
| FIT501 | 0.011067 | 0.012793 | 0.010428 | BiGRU |
| FIT502 | 0.014464 | 0.014014 | 0.013822 | BiGRU |
| FIT503 | 0.00605 | 0.006301 | 0.00513 | BiGRU |
| FIT504 | 0.004668 | 0.00476 | 0.003971 | BiGRU |
| P501 | 0.003856 | 0.005784 | 0.003097 | BiGRU |
| PIT501 | 1.743642 | 2.028927 | 1.375167 | BiGRU |
| PIT502 | 0.107684 | 0.090888 | 0.07483 | BiGRU |
| PIT503 | 1.348192 | 1.441471 | 1.046558 | BiGRU |
| FIT601 | 0.020084 | 0.02018 | 0.020002 | BiGRU |
| P602 | 0.011472 | 0.011745 | 0.011743 | LSTM |

*2) Ensemble Model 2: Highest Percentage of Values Within Tolerance (PWT)*

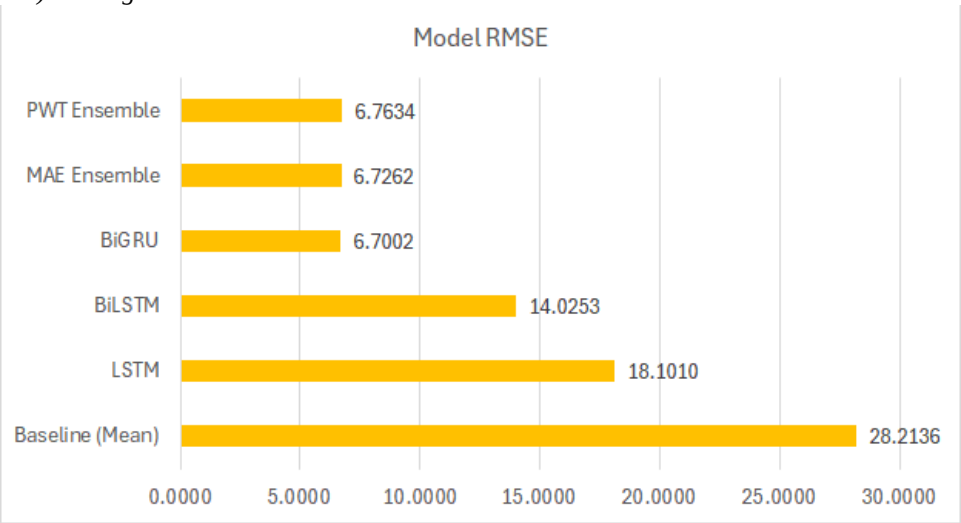
| Feature | LSTM PWT (%) | BiLSTM PWT (%) | BiGRU PWT (%) | Selected Model |
|---|---|---|---|---|
| FIT101 | 81.057 | 73.449 | 82.719 | BiGRU |
| LIT101 | 82.592 | 90.905 | 98.330 | BiGRU |
| MV101 | 95.224 | 94.359 | 95.226 | BiGRU |
| P101 | 95.679 | 95.693 | 95.775 | BiGRU |
| P102 | 99.896 | 99.896 | 99.896 | Any |
| AIT201 | 99.246 | 98.994 | 98.581 | LSTM |
| AIT202 | 100.000 | 100.000 | 100.000 | Any |
| AIT203 | 99.497 | 100.000 | 100.000 | BiLSTM or BiGRU |
| FIT201 | 77.752 | 77.175 | 84.013 | BiGRU |
| MV201 | 94.450 | 94.491 | 94.758 | BiGRU |
| P201 | 95.979 | 95.820 | 95.978 | LSTM |
| P203 | 96.112 | 96.149 | 95.630 | BiLSTM |
| P204 | 100.000 | 100.000 | 100.000 | Any |
| P205 | 96.116 | 96.149 | 96.003 | BiLSTM |
| P206 | 100.000 | 100.000 | 100.000 | Any |
| DPIT301 | 88.783 | 87.957 | 90.610 | BiGRU |
| FIT301 | 86.032 | 84.917 | 89.459 | BiGRU |
| LIT301 | 96.052 | 96.362 | 99.357 | BiGRU |
| MV301 | 98.305 | 98.305 | 98.305 | Any |
| MV302 | 94.351 | 94.890 | 95.521 | BiGRU |
| MV303 | 96.183 | 95.828 | 97.520 | BiGRU |
| MV304 | 95.695 | 94.655 | 96.438 | BiGRU |
| P301 | 100.000 | 100.000 | 100.000 | Any |
| P302 | 94.954 | 94.642 | 95.269 | BiGRU |
| AIT401 | 99.747 | 100.000 | 99.734 | BiLSTM |
| AIT402 | 99.160 | 99.215 | 99.272 | BiGRU |
| FIT401 | 97.951 | 97.950 | 97.947 | LSTM |
| LIT401 | 98.534 | 99.417 | 100.000 | BiGRU |
| P402 | 100.000 | 100.000 | 100.000 | Any |
| P403 | 100.000 | 100.000 | 100.000 | Any |
| UV401 | 99.691 | 99.691 | 99.790 | BiGRU |
| AIT501 | 100.000 | 100.000 | 100.000 | Any |
| AIT502 | 99.217 | 99.205 | 99.308 | BiGRU |
| AIT503 | 100.000 | 100.000 | 100.000 | Any |
| AIT504 | 98.747 | 98.097 | 97.271 | LSTM |
| FIT501 | 98.759 | 98.758 | 99.000 | BiGRU |
| FIT502 | 99.501 | 99.501 | 99.736 | BiGRU |
| FIT503 | 98.752 | 98.751 | 98.869 | BiGRU |
| FIT504 | 98.746 | 98.745 | 98.731 | LSTM |
| P501 | 99.691 | 99.691 | 99.790 | BiGRU |
| PIT501 | 98.756 | 98.755 | 98.997 | BiGRU |
| PIT502 | 69.927 | 87.937 | 90.490 | BiGRU |
| PIT503 | 98.760 | 98.759 | 99.002 | BiGRU |
| FIT601 | 76.211 | 76.733 | 48.842 | BiLSTM |
| P602 | 98.903 | 98.903 | 98.903 | Any |

*D. Plots of Comparison of Ensemble Models with Individual Models*

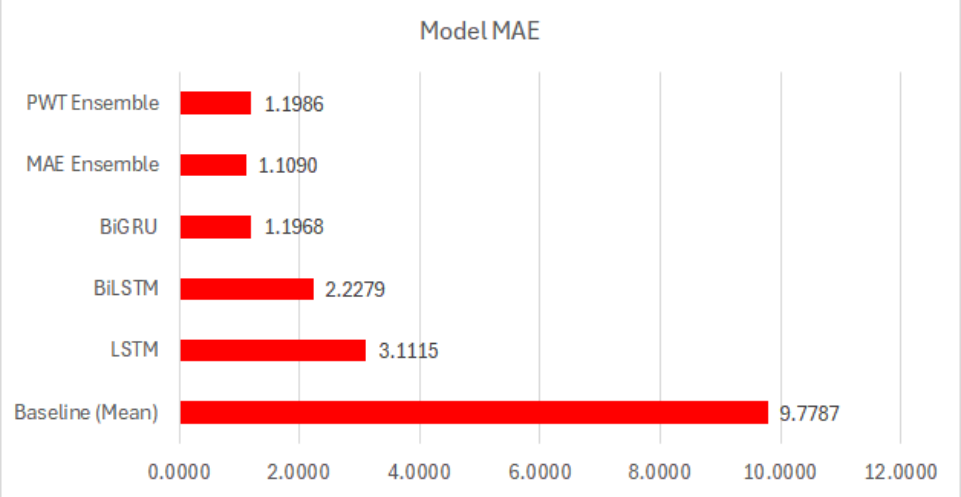
*1) Using MSE*



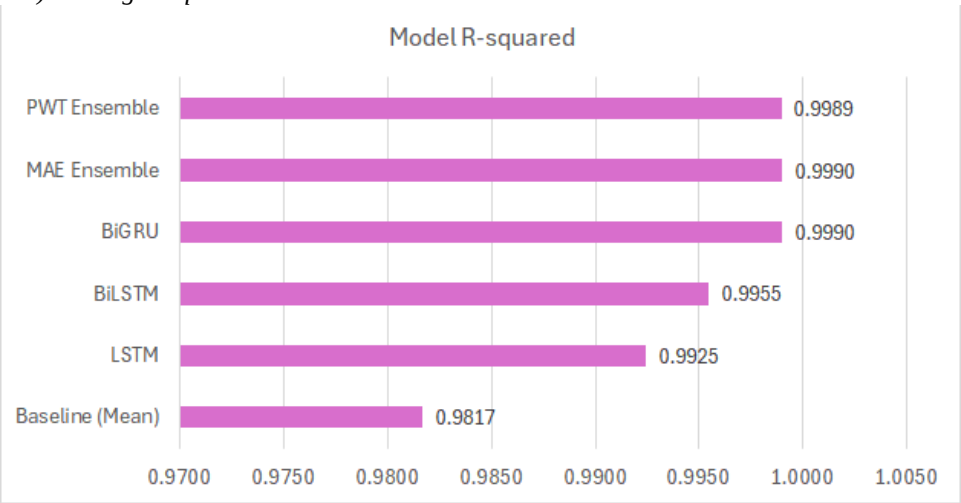

*2) Using RMSE*




*3) Using MAE*



4) *Using R-squared*




5) *Using PWT*