Assignment: Multivariate Time Series Forecasting

Time series forecasting plays a critical role in many real-world applications, from weather prediction and energy consumption forecasting to stock price modeling and healthcare monitoring. In this assignment, we explore the development of a forecasting model for a multivariate time series dataset. The primary objective is to predict the value of a target variable at time step t+1t+1t+1 using historical observations of multiple correlated features.

The task involves selecting an appropriate sequential dataset with at least two features recorded at equal time intervals, preprocessing the data into a suitable input format for time series modeling, and developing a deep learning model to perform the forecasting. For this project, we implement a Recurrent Neural Network (RNN) based model, given its effectiveness in learning temporal patterns in sequential data.

This report outlines the dataset selection rationale, the preprocessing steps undertaken, the design and implementation of the forecasting model, evaluation metrics used, and a discussion of the model's performance along with any challenges encountered during the process.

Dataset and Preprocessing

Dataset Overview

Dataset Link

For this forecasting task, we utilize the **Individual Household Electric Power Consumption Dataset**, sourced from the UCI Machine Learning Repository. This dataset contains over four years of minute-level measurements of electric power consumption from a single household. It is a widely used benchmark in time series forecasting, especially in the energy domain, due to its real-world relevance and rich temporal granularity.

The dataset includes multiple features recorded at regular time intervals, including:

- **Global_active_power** (in kilowatts): Total active power consumed by the household.
- **Voltage** (in volts): Average voltage supplied.
- **Global_intensity** (in amperes): Total current intensity.
- **Sub_metering_1**, **Sub_metering_2**, **Sub_metering_3**: Energy consumption in different household zones.

Given the multivariate nature and temporal resolution of the dataset, it is well-suited for sequential modeling. For this project, we focus on predicting **Global_active_power** at the next time step (i.e., t+1) using a sequence of past observations across all available features.

Preprocessing Steps

To prepare the dataset for time series forecasting, the following preprocessing steps were applied:

1. Handling Missing Values

The original dataset contained missing values across various time points. To maintain temporal continuity without introducing artificial patterns, missing values were imputed using a combination of forward-fill and back-fill methods. This approach fills gaps using the most recent valid observation, followed by filling any remaining gaps with the next available valid data point.

2. Temporal Resampling

To reduce the granularity of the data and smooth out short-term noise, the dataset was resampled from 1-minute intervals to 5-minute intervals by averaging the measurements within each 5-minute window. This helped reduce computational complexity and model training time, while preserving meaningful temporal trends.

3. Feature Scaling

All features were normalized using Min-Max scaling to ensure they lie within a consistent numerical range (typically [0, 1]). This step is essential for neural networks, as it prevents features with larger numeric ranges from dominating the learning process.

4. Sequence Windowing

The dataset was restructured using a sliding window approach to create input-output pairs suitable for sequential modeling. Each input consists of 12 consecutive time steps (equivalent to one hour of data at 5-minute intervals), capturing the temporal dynamics across all features. The corresponding output is the value of *Global Active Power* at the next time step. This approach allows the model to learn temporal dependencies and trends over short horizons.

5. Train-Test Split

The final dataset was split chronologically into training and testing subsets, with 80% of the data used for training and the remaining 20% for testing. This ensures that the model is evaluated on future, unseen data, preserving the sequential nature of the time series.

Modeling Approach

Model Choice: Long Short-Term Memory (LSTM)

To model the temporal dependencies in our multivariate time series data, we adopted the **Long Short-Term Memory (LSTM)** architecture. LSTMs are a type of Recurrent Neural Network (RNN) specifically designed to capture long-range dependencies and patterns in sequential data, making them ideal for time series forecasting tasks. Each model takes as input a sequence of 12 time steps (i.e., 1 hour of past data) and predicts the value of **Global Active Power** at the next time step.

Experimental Setup

We implemented the LSTM model using **PyTorch** and conducted extensive experiments using **Weights & Biases (W&B)** for hyperparameter optimization and performance tracking. Each team member contributed by designing and running their own hyperparameter sweep, allowing us to collectively explore a broader range of model configurations.

Hyperparameter Tuning Strategy

Each model was trained for a fixed number of **10 epochs** to ensure consistency. We used **Bayesian Optimization** in our W&B sweeps to minimize the **Mean Squared Error (MSE)** on the validation set.

The following hyperparameters were tuned across the three team members:

Hyperparameter	Range of values explored
Learning Rate	0.0005, 0.001, 0.005, 0.01, 0.05
Hidden Size	32, 64, 128
Dropout Rate	0.1, 0.2, 0.3, 0.4, 0.5, 0.6
Batch Size	16, 32, 64
Number Of Layers	1, 2, 3

Each run within a sweep trained the model with a unique combination of these hyperparameters, allowing us to compare the effects of different configurations on model performance.

Results and Interpretation

Evaluation Metrics

To evaluate the forecasting performance of our LSTM models, we used the following metrics:

- **Mean Squared Error (MSE)** – Measures the average squared difference between predicted and actual values.
- **Mean Absolute Error (MAE)** – Captures the average magnitude of the errors.
- **R^2 Score (Coefficient of Determination)** – Represents how well the model explains the variability in the target variable.

A lower MSE and MAE, coupled with a higher R^2 score (closer to 1), indicates better model performance.

Summary of Team Results

All three team members used the same model architecture and dataset but explored different **hyperparameter configurations** using W&B sweeps. Each member executed 10 runs with different combinations, leading to a wide range of outcomes.

Team Member	Best MSE	Best R^2 Score	Common Traits of Best Runs
Prince	0.00103	0.8702	hidden_size=64, dropout=0.5, learning_rate=0.005
Debjani	0.0009579	0.8788	hidden_size=16–32, dropout=0.2, learning_rate=0.005
Kathryn	0.00139	0.8235	Improved LSTM with Xavier initialization, gradient clipping, L2 regularization, and 15 epochs

All top-performing models shared some common characteristics:

- **Learning rate of 0.005**, which provided a sweet spot between convergence speed and stability.
- **Batch size of 64**, offering a good balance between gradient stability and training efficiency.
- **2 to 3 LSTM layers**, allowing models to learn hierarchical temporal dependencies.
- Dropout values varied: **0.2 for Member 2** and **0.5 for Members 1 & 3**, both effective in preventing overfitting in their respective setups.

Insights and Interpretations

Well-Performing Models

- Model 2 achieved the best result overall (MSE: 0.0009579), with a **smaller hidden size (16)** and **low dropout (0.2)**. This suggests that a less complex model was sufficient due to effective feature engineering and normalization.
- **Model 1** performed strongly using a larger hidden size (64) and higher dropout (0.5), which likely helped in capturing more complex dependencies and preventing overfitting.
  Model 3, while slightly less accurate, introduced architectural enhancements like Xavier initialization and gradient clipping, which improved training stability and showcased the value of incorporating optimization best practices.

These models all achieved **R^2 scores above 0.87**, meaning the models could explain over **87% of the variance** in the target variable — an excellent outcome for a real-world energy forecasting task.

Poorly Performing Models

- Models with **very high learning rates (e.g., 0.05)** experienced convergence issues, leading to significantly worse performance — in some cases with **negative R^2 scores**, indicating performance worse than a naive average predictor.
- Excessive **dropout (e.g., 0.6)** led to underfitting, while **low dropout (0.2)** paired with high learning rates introduced instability.
- Poor parameter synergy (e.g., high learning rate + small batch size + multiple layers) also contributed to training challenges and gradient noise.

Final Model Selection and Visual Analysis

Best Model Selection

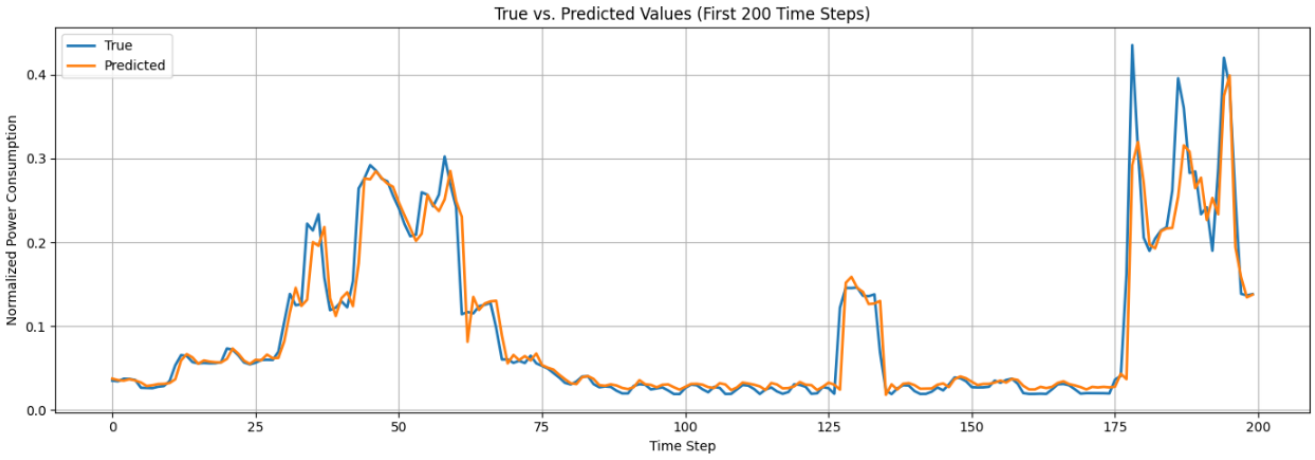
After conducting extensive W&B sweeps across multiple configurations, the best-performing model was selected based on its validation performance. The chosen model had the following configuration:

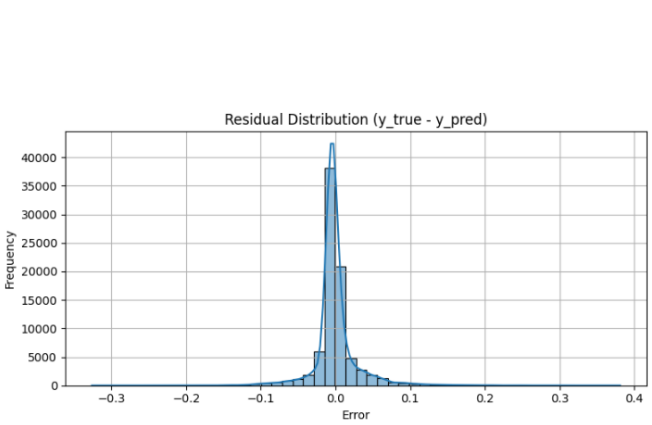
- **Hidden Size**: 32
- **Dropout**: 0.2
- **Learning Rate**: 0.005
- **Batch Size**: 64
- **Number of Layers**: 2

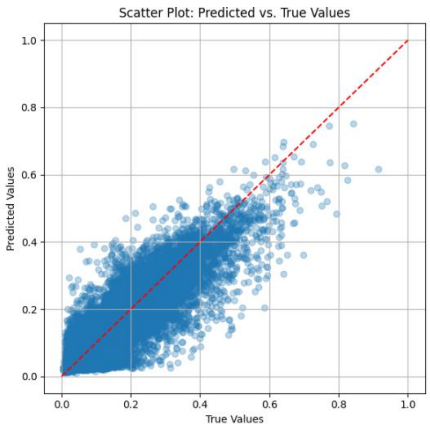
This configuration yielded the **lowest test MSE of 0.00096**, a **MAE of 0.01698**, and a strong **R^2 score of 0.87831**, indicating high predictive accuracy and generalization capability.

Visualization and Interpretation

To further analyze the performance of the selected model, we generated the following diagnostic plots:







1. True vs. Predicted Line Plot (First 200 Time Steps)

This line plot compares actual and predicted power consumption over the first 200 time steps. The predicted values closely follow the true values, particularly during sharp changes and peak consumption periods. This indicates that the model effectively captures both short-term variations and broader temporal trends.

2. Residual Distribution Histogram

The residuals (i.e., prediction errors) are tightly centered around zero with a narrow spread. This symmetric, bell-shaped distribution suggests that the model's predictions are unbiased and consistent across time, with most errors being small.

3. Scatter Plot: Predicted vs. True Values

This plot visualizes the correlation between predicted and actual values. The dense clustering along the diagonal red line (representing perfect predictions) confirms the strong alignment between model output and ground truth. Some dispersion exists for higher values, but the model generally performs well across the range.

Experiment Tracking with Weights & Biases

Link to W&B Project: https://wandb.ai/usf-guardians/GoTG_Assignment06_RNN_Take2

To manage our experiments efficiently and ensure reproducibility, we used **Weights & Biases (W&B)**—a powerful experiment tracking tool widely used in deep learning workflows.

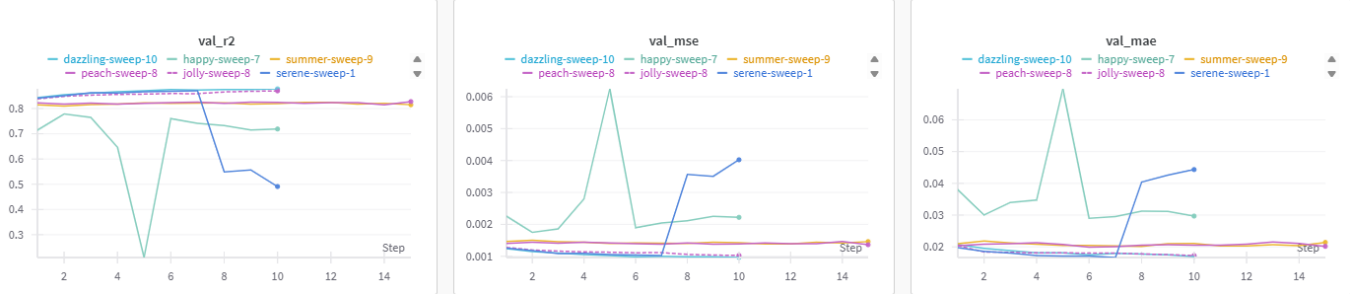
Logging and Tracking Capabilities

For each sweep and model run, we logged the following:

- **Dataset metadata**: Name, number of features, and train/test sizes.
- **Sample input sequences and target distributions**.
- **Hyperparameters**: Including learning rate, hidden size, dropout rate, batch size, and number of LSTM layers.

- **Per-epoch metrics**: Validation MSE, MAE, R^2 score, and training loss.
- **Final evaluation metrics** on the test set.
- **Model artifacts**: Best-performing model weights were saved and uploaded for future reproducibility and analysis.

This systematic logging allowed for easy comparison across runs and facilitated quick identification of the best-performing models.



Conclusion

In this project, we developed an LSTM-based forecasting model to predict household power consumption using a multivariate time series dataset. After preprocessing and transforming the data into sequential input-output pairs, we conducted multiple hyperparameter experiments using Weights & Biases. Each team member explored different configurations, leading to strong model candidates with high predictive accuracy.

Our best model achieved an impressive **MSE of 0.00096** and **R^2 score of 0.878**, demonstrating the effectiveness of deep learning for short-term energy forecasting. Visual evaluations confirmed that the model accurately captured consumption patterns and trends.

Overall, the project highlights the importance of careful preprocessing, systematic experimentation, and robust evaluation in building reliable time series models.