Time Series

A **Recurrent Neural Network (RNN)** is a type of **artificial neural network** that is designed to recognize **patterns in sequences of data** — such as time series, text, speech, or video frames.

Think of an **RNN** as a person with **short-term memory** trying to make predictions or decisions based on **past experiences**.

For example:

- You're trying to guess **tomorrow's weather**.
- You don't just look at today — you remember it was raining yesterday, cloudy today, and that helps you guess what might come next.

That's exactly what an RNN does with data.
It **remembers what happened before** to make better predictions about what's coming.

Problem Statement:

Develop and implement a Recurrent Neural Network (RNN) model to forecast future values of a time-series dataset. The dataset contains historical observations recorded at consistent time intervals. The goal is to train the RNN to capture underlying patterns and dependencies in the time-series data, enabling accurate prediction of upcoming values.

Introduction:

Predicting future values in time series data poses significant challenges, particularly due to the complex patterns and temporal dependencies involved. Traditional neural network models often fail to effectively capture these time-based relationships. In this study, we employ advanced deep learning architectures—such as LSTM, GRU, Simple RNN, and Conv1D—to improve the accuracy of time series forecasting. The dataset utilized comprises weather observations from Germany recorded between 2009 and 2016, including temperature, pressure, and humidity readings taken at 10-minute intervals. By training these models to forecast future temperatures, we aim to demonstrate their effectiveness in handling time series prediction tasks.

Preprocessing:

Typically, raw time series data undergo preprocessing before being used to train deep learning models. Because the dataset's features exist on varying scales, normalization is required to ensure consistency across all inputs. The mean and standard deviation are computed from a chosen subset of the data (specifically, the first 200,000+ timesteps), and

each feature is individually normalized using these values. This normalization step helps promote faster convergence and improved stability during the training process.

Data Preparation:

To make the dataset suitable for supervised learning, input-output pairs are generated by creating sequences and their corresponding target values. A custom data windowing technique is applied, where sliding windows of historical data (for example, the previous five days) are extracted and matched with the temperature reading 24 hours ahead. These samples are generated dynamically to optimize memory usage while maintaining a steady data flow during training.

This process is efficiently handled using Keras's timeseries_dataset_from_array() utility.

Dataset Parameters:

Three separate datasets are prepared and divided into training, validation, and testing sets. The time series data are resampled to one data point per hour. Each input sequence spans 120 hours (equivalent to 5 days), and the corresponding target is set to the temperature 24 hours beyond the sequence's endpoint. This configuration enables the model to learn patterns useful for producing medium-range temperature forecasts based on recent trends.

Baseline Approach:

A basic baseline model is established for performance comparison, using the simple assumption that the temperature at the same time tomorrow will be identical to the current temperature. While straightforward, this approach provides a reference point, achieving a validation Mean Absolute Error (MAE) of 2.44°C and a test MAE of 2.62°C. This benchmark allows the performance of more advanced models to be effectively evaluated.

Deep Learning Techniques:

To enhance forecasting accuracy, deep learning-based approaches are applied. The foundation of these methods lies in Recurrent Neural Networks (RNNs), which are designed to capture temporal dependencies within sequential data. This study explores several advanced RNN architectures, including Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks, which are capable of retaining information over longer time horizons while addressing challenges like the vanishing gradient problem. Additionally, 1-Dimensional Convolutional Neural Networks (Conv1D) and hybrid models that combine Conv1D with RNNs are investigated, highlighting the versatility and effectiveness of deep learning for time series forecasting.

Model	Dense Unit	Dropout	Validation MAE	Test MAE
Basic Model	16	No	3.78	3.57
LSTM with 3 Layers	8	No	0.26	0.24
LSTM with 3 Layers	16	No	0.25	0.26
LSTM with 3 Layers	32	No	0.24	0.26
LSTM with 4 Layers	16	No	0.23	0.26
GRU with 3 Layers	8	No	0.27	0.26
GRU with 3 Layers	16	No	0.23	0.25
GRU with 3 Layers	32	No	0.26	0.26
GRU with 4 Layers	16	No	0.25	0.27
1D Convolution and LSTM	32	No	0.29	0.25
1D Convolution and Simple RNN	32	No	0.24	0.27
1D Convolution and GRU	32	No	0.23	0.25
1D Convolution, LSTM and Dropout	32	Dropout 0.5	0.27	0.27
1D Convolution, Simple RNN and Dropout	16	Dropout 0.5	0.29	0.27
1D Convolution, GRU and Dropout	32	Dropout 0.5	0.26	0.24
Bi Directional GRU and Dropout	16	Dropout 0.5	0.24	0.25

Model Performance Analysis

Basic Machine Learning Model

- Consists of a simple dense (feedforward) network with 16 units and no dropout.
- Represents a traditional model without any sequential learning ability (no RNN or CNN layers).
- Validation MAE: 3.78 | Test MAE: 3.57
- Performance is significantly worse than all deep learning models.
- Indicates poor capability to learn complex or temporal patterns in the data.
- Serves mainly as a baseline, confirming that advanced models are necessary for effective sequence modeling.

LSTM (Long Short-Term Memory) Models

- Designed to handle long-term dependencies in sequential data.
- Several configurations tested:
    - 3-layer LSTMs with 8, 16, and 32 dense units.
    - 4-layer LSTM with 16 dense units.
- Performance summary:
    - Validation MAE: 0.23–0.26
    - Test MAE: 0.24–0.26
- Best model: 4-layer LSTM with 16 units (Validation MAE = 0.23, Test MAE = 0.26).
- Increasing the number of layers slightly improved validation accuracy but did not greatly affect test performance.
- Adding convolution (1D Conv + LSTM) or dropout (0.5 rate) did not improve results notably.
- Suggests that standard LSTM structures already captured the data's sequential patterns well.
- Demonstrates stable performance but not as efficient as GRU in generalization.

GRU (Gated Recurrent Unit) Models

- A simplified recurrent architecture compared to LSTM, using fewer gates.
- Tested with 3 and 4 layers and dense units of 8, 16, and 32.
- Performance summary:
    - Validation MAE: 0.23–0.27
    - Test MAE: 0.25–0.27
- Best model: 3-layer GRU with 16 units (Validation MAE = 0.23, Test MAE = 0.25).
- Increasing units or layers beyond this did not improve results, sometimes even slightly worsened them (possible overfitting).
- 1D Conv + GRU hybrid performed equivalently well (Validation MAE = 0.23, Test MAE = 0.25).
- 1D Conv + GRU + Dropout showed strongest generalization (Validation MAE = 0.26, Test MAE = 0.24).
- Bidirectional GRU with Dropout (16 units) produced balanced performance (Validation MAE = 0.24, Test MAE = 0.25).
- GRUs overall delivered the best trade-off between training accuracy and test generalization.
- Conclusion: GRUs outperform LSTMs slightly while using simpler, faster architectures.

1D Convolutional and Hybrid Models

- Combine 1D convolutional layers (for local pattern extraction) with recurrent layers (for temporal learning).
- Tested hybrids:
    - Conv + LSTM
    - Conv + Simple RNN
    - Conv + GRU
    - Plus variants with dropout (0.5 rate).
- Performance summary:

    - Conv + GRU: 0.23 / 0.25 (very strong results).
    - Conv + Simple RNN: 0.24 / 0.27 (competitive, but less accurate).
    - Conv + LSTM: 0.29 / 0.25 (weaker validation, solid test performance).
- Dropout slightly improved generalization in some cases:
    - Conv + GRU + Dropout achieved lowest test MAE (0.24).
- Indicates that CNN layers help extract useful features, especially when paired with GRUs.
- Overall, hybrid models perform nearly as well as the best GRU networks, often with better robustness.

Models with Dropout (0.5 rate)

- Dropout used to reduce overfitting by randomly deactivating neurons during training.
- Applied to hybrid and bidirectional models.
- Impact:
    - Slightly increased validation error in most models.
    - Helped maintain or improve test MAE, suggesting better generalization.
- Best dropout result: Conv + GRU + Dropout (Validation MAE = 0.26, Test MAE = 0.24).
- Demonstrates that dropout improves robustness even when the models are already well-trained.

Bidirectional GRU with Dropout

- Processes sequences in both directions, providing full temporal context.
- Configuration: 16 dense units, dropout = 0.5.
- Validation MAE: 0.24 | Test MAE: 0.25
- Not the absolute best in validation, but one of the most stable and generalizable across both sets.
- Ideal for applications requiring context from both past and future sequence elements.

Performance Summary

- Best validation accuracy:
    - GRU (3 layers, 16 units) → Validation MAE = 0.23, Test MAE = 0.25
- Best test generalization:
    - Conv + GRU + Dropout → Validation MAE = 0.26, Test MAE = 0.24
- Other strong performers:
    - 4-layer LSTM (16 units): 0.23 / 0.26
    - Bidirectional GRU with Dropout: 0.24 / 0.25
- Key insight:
    - GRU models are the most efficient and best performing overall.
    - Hybrid Conv + GRU models show excellent generalization and may be the optimal practical choice.

Conclusion:

To conclude, the experimental comparison of different neural network architectures reveals clear distinctions in their performance based on Validation and Test MAE results. The basic dense model recorded a significantly higher error (Validation MAE = 3.78, Test MAE = 3.57), confirming that simple feedforward networks are not suitable for capturing temporal or sequential dependencies in the data.

Among the advanced architectures, both LSTM and GRU models demonstrated strong capability in learning sequential relationships, with error values reduced to below 0.30. The LSTM with four layers and 16 dense units achieved a Validation MAE of 0.23 and a Test MAE of 0.26, showing solid training performance but moderate generalization.

The GRU-based models generally outperformed the LSTMs, offering slightly lower validation and test errors with fewer parameters. In particular, the three-layer GRU with 16 dense units achieved the lowest Validation MAE of 0.23 and a Test MAE of 0.25, making it the best model in terms of training and validation accuracy.

Hybrid models that integrated 1D convolutional layers with recurrent units also performed competitively. Among these, the 1D Convolution + GRU model with a 0.5 dropout rate achieved the lowest Test MAE of 0.24, indicating superior generalization and robustness on unseen data. The combination of convolutional feature extraction, recurrent sequence

modeling, and dropout regularization helped this hybrid network balance accuracy and stability effectively.

Overall, it can be concluded that:

- The GRU (3 layers, 16 units) model is the best performer during validation, achieving the lowest MAE (0.23).
- The 1D Convolution + GRU + Dropout model demonstrates the strongest generalization, with the lowest test MAE (0.24).
- LSTM models are reliable but slightly less efficient, while hybrid and GRU-based architectures consistently offer better performance and robustness.
- Therefore, for real-time or production applications where adaptability and stability are crucial, the 1D Convolution + GRU with dropout architecture represents the optimal balance between accuracy and generalization