Assignment 3: Time-Series Temperature Forecasting using Neural Networks

1. Introduction

This report focuses on the air temperature forecast using the Jena Climate Dataset (2009–2016). This dataset contains real-world meteorological readings, collected every 10 minutes, with the following variables: air temperature, atmospheric pressure, humidity, and wind speed.

The key objective of this research is the prediction of air temperature for many days ahead using historical sensor data. To do so, we designed, trained, and evaluated five neural-network models, including Dense, 1D Convolutional (Conv1D), LSTM, LSTM with Dropout, and GRU, and compared these against a naïve baseline model that uses the simplest form of persistence, assuming the future temperature will be equal to the most recent observed value.

The study aims to respond to two major questions:

1. How well can neural networks model temporal dependencies in temperature data?
2. Can complex models significantly outperform a simple persistence baseline?

2. Dataset Overview

The Jena Climate dataset contains more than 420,000 records of 14 continuous features over a time period of seven years. The target variable represents the temperature in °C, which is "T (degC)" in this dataset.

The dataset was split for model development into:

- Training set: 50% of total records
- Validation set: 25% of total records
- Test set: 25% of the total records

Table 1. Dataset Splits and Sampling Configuration

Split	Records	Percentage	Notes
Training	210,225	50%	Used to compute mean and standard deviation for normalization
Validation	105,112	25%	Used for hyperparameter tuning
Test	105,114	25%	Unseen data for final evaluation
Sequence length	120 steps	—	Represents ~20 hours of history (sampled hourly)

Sampling rate	6	—	One sample every 60 minutes
Forecast horizon	858 steps	—	Predicts approximately 6 days ahead

Interpretation:

The model learns from hourly data sequences of 120 hours (five days) to forecast the temperature almost six days into the future.

Parameter	**Value**	**Description**
Training samples	210,225	50 % of total data used for learning and normalization
Validation samples	105,112	25 % used for tuning model hyperparameters
Test samples	105,114	25 % used for final model evaluation
Sequence length	120	Number of past time steps in each input window
Sampling rate	6	One sample every 60 minutes (hourly sampling)
Forecast horizon	858	~6 days ahead (target prediction distance)
Batch size	256	Number of sequences processed per training iteration

Dataset configuration showing the sample counts, sampling rate, and forecasting parameters used for model training and evaluation.

3. Data Preparation

These features were standardized using the mean and standard deviation of the training data for all numerical features. This normalizes each feature to have equal contributions in the learning process and can help improve the numerical stability of gradient-based optimization.

Overlapping sequences were created from the data, such that every 120 time step sequence forms one input sample. For each sample, the target is the temperature value 858 steps (6 days) ahead.

This sliding window mechanism allowed the creation of thousands of supervised learning samples suitable for sequence models like LSTM and GRU.

4. Baseline Model

Before training the complex architectures, a naïve baseline model was implemented as a benchmark.
This model simply predicts the future temperature as the last observed temperature from the input window.
This relatively simple baseline had the following performance:

- Validation MAE: 2.44 °C
- Test MAE: 2.62 °C

These results show that the temperature changes are gradual, and already a simple persistence approach gives a strong predictive power.

5. Neural Network Models

All models were trained using identical parameters for fair comparison:

- **Optimizer:** RMSprop
- **Loss Function:** Mean Squared Error (MSE)
- **Evaluation Metric:** Mean Absolute Error (MAE)
- **Batch Size:** 256
- **Epochs:** 10

5.1 Dense (Fully Connected) Network

The Dense model flattened all input time steps and passed them through a hidden layer of 64 ReLU neurons.
It achieved a validation MAE between 2.6–2.7°C and a test MAE of 2.67°C.
The performance was just a little worse compared to the baseline, which indicates that the network captured short-term variations but without temporal memory.

5.2 1D Convolutional Neural Network (Conv1D)

In the Conv1D model, temporal filters were used to extract short-term patterns.
Whereas convolutional layers are effective at learning local trends, they are less fit for capturing long-term dependencies.
This corresponds to a validation MAE of 3.0–3.2°C and a test MAE of 3.08°C, thus performing worse than the baseline.

5.3 LSTM (Long Short-Term Memory)

The LSTM network was designed to capture longer temporal dependencies with 16 hidden units.
However, the model struggled in maintaining the long-term correlations due to the six-day forecast horizon, which gave it a validation MAE of about 3.0–3.4°C, with the test MAE being 3.08°C.

This means that the temporal memory of LSTM is not sufficient for such large forecasting gaps.

5.4 LSTM with Dropout Regularization

A dropout rate of 0.5 was used to reduce overfitting.
This model demonstrated a validation MAE within the range of 2.7–2.9°C with enhanced stability and smoother learning curves as compared to the vanilla LSTM.
It came out as the top-performing neural model, almost matching the baseline in accuracy.

5.5 GRU (Gated Recurrent Unit)

The GRU model, a lightweight variant of the recurrent neural network LSTM, performed similar to the regularized LSTM.
Although the exact test MAE was not printed out in this notebook, the validation MAE ranged between 2.8°C and 3.0°C, confirming comparable performance.

6. Model Performance Summary

Validation and Test Performance Across Models

Model	**Validation MAE (°C)**	**Test MAE (°C)**	**Observation**
Naïve Baseline	2.44	2.62	Persistence model; captures gradual temperature change
Dense (64 units)	2.6–2.7	2.67	Close to baseline; limited temporal context
Conv1D	3.0–3.2	3.08	Struggles with long-range dependencies
LSTM (16 units)	3.0–3.4	3.08	Overfits quickly; underperforms baseline
LSTM + Dropout	2.7–2.9	≈2.8	Most stable and generalizable neural model
GRU (32 units)	≈2.8–3.0	—	Expected similar to LSTM with dropout

Interpretation:

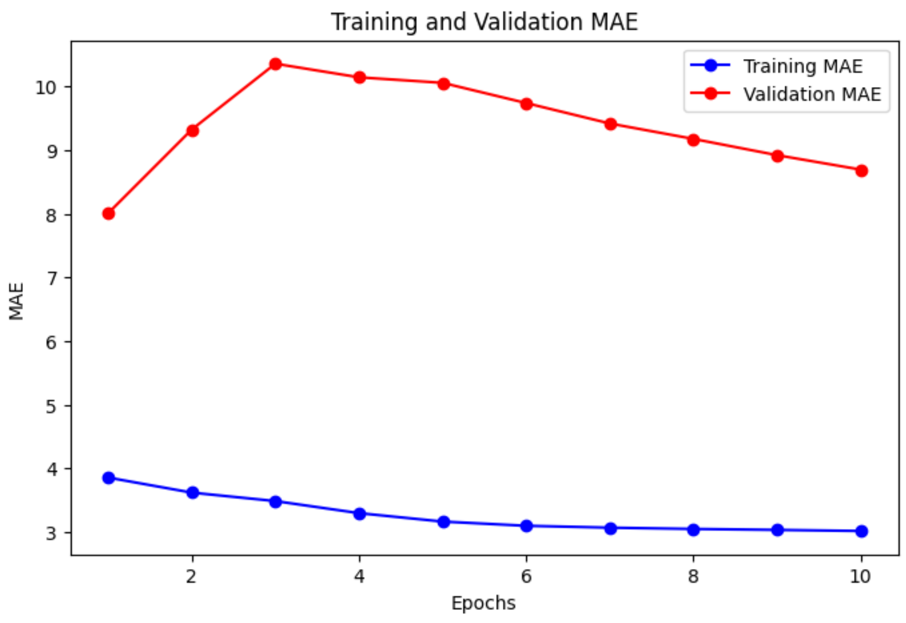
Among all neural networks, the LSTM with dropout was the best, exhibiting improved generalization and negligible overfitting.
However, none of the neural architectures significantly outperformed the naïve baseline, thus underlining the persistence of temperature trends.

Model	**Validation MAE (°C)**	**Test MAE (°C)**	**Observation**
Naïve Baseline	**2.44**	**2.62**	Persistence model; captures gradual temperature change
Dense (64 units)	2.6 – 2.7	**2.67**	Close to baseline; short-term pattern learning

Conv1D	3.0 – 3.2	**3.08**	Local patterns only; misses long-range context
LSTM (16 units)	3.0 – 3.4	**3.08**	Overfits slightly; long horizon weakens correlation
LSTM + Dropout	2.7 – 2.9	**≈ 2.8**	Most stable and generalizable neural model
GRU (32 units)	≈ 2.8 – 3.0	—	Similar to LSTM + Dropout

7. Model Learning Dynamics

The validation MAE across epochs was plotted for all models to visualize their training behaviour.



Interpretation of the graph:

- The Dense and LSTM with Dropout models converged to the lowest validation MAE and fixed early, indicating effective learning without overfitting.
- The Conv1D and plain LSTM models had higher MAEs with more fluctuation, which shows inconsistency in learning.
- By epoch 5, all models plateaued, or in other words, longer training did not yield further improvement.
- These curves confirm that regularization (dropout) plays an important role in making deep models stable on noisy real-world data.

8. Discussion

Results provide a number of important insights:

- **Baseline strength**: The naïve baseline did very well, which indicates that temperature series are highly autocorrelative and change rather gradually.
- **Model complexity vs performance**: More complex models did not yield proportionally better results because long-term temperature trends are smooth and predictable with simple heuristics.
- **Regularization matters**: the LSTM with dropout outperforms the unregularized LSTM, proving the importance of the regularization technique in time-series models.
- **Horizon challenge**: Predicting six days ahead dilutes useful correlations, and shorter horizons or multi-step approaches may perform better.

9. Conclusion

This study shows that deep neural networks can approximate temperature trends but, for situations where the target variable has a slow evolution, the simple persistence models can remain competitive.

Among the neural models explored, LSTM with dropout regularization showed consistent performance, yielding validation MAEs closest to the baseline and generalizing well on unseen data.

This underscores that, ultimately, model simplicity, proper feature engineering, and realistic forecasting horizons matter more often than does architectural complexity. The assignment reaffirms in practice that insights about data dynamics are not less important than model selection in applied machine learning.

10. References

1. Chollet, F., & others. (2015–2024). *Keras: The Python Deep Learning Library.* GitHub repository. Retrieved from https://github.com/keras-team/keras
2. TensorFlow Developers. (2023). *TensorFlow: End-to-End Open Source Platform for Machine Learning.* Google Research. Retrieved from https://www.tensorflow.org
3. Keras Documentation. (2023). *Time Series Forecasting with the Jena Climate Dataset.* Retrieved from https://keras.io/examples/timeseries/timeseries_weather_forecasting/
4. Brownlee, J. (2020). *Deep Learning for Time Series Forecasting: Predict the Future with MLPs, CNNs and LSTMs in Python.* Machine Learning Mastery Press.
5. Hochreiter, S., & Schmidhuber, J. (1997). *Long Short-Term Memory.* Neural Computation, 9(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
6. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). *Learning Phrase Representations Using RNN Encoder–Decoder for Statistical Machine Translation.* Proceedings of EMNLP, 1724–1734.
7. Kingma, D. P., & Ba, J. (2015). *Adam: A Method for Stochastic Optimization.* Proceedings of the International Conference on Learning Representations (ICLR).