AML Assignment 3: Deep Learning for Time Series Forecasting on the Jena Climate Dataset

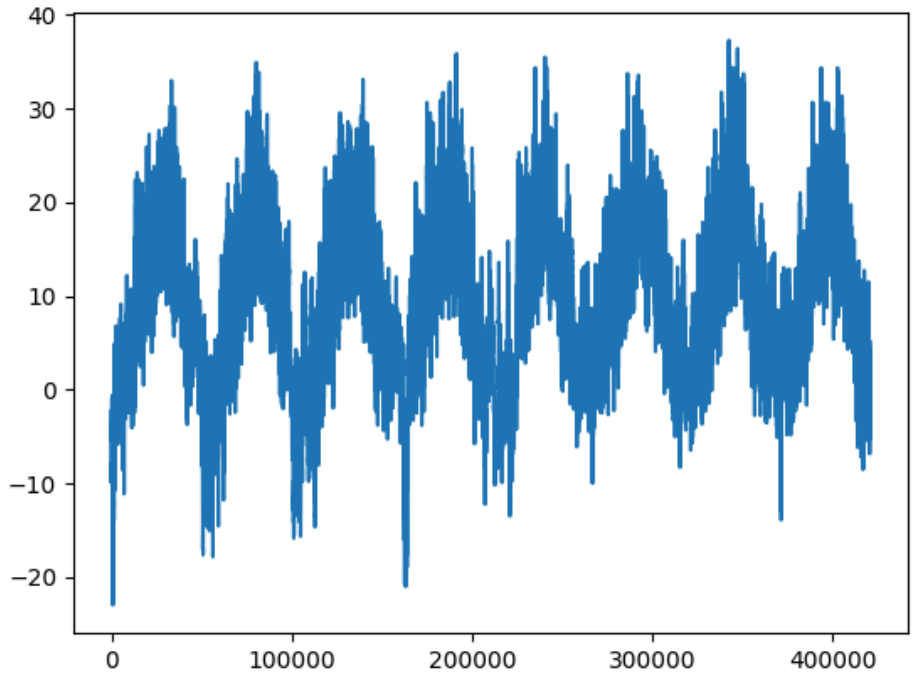
Introduction

This activity delves into the application of deep learning models to weather time series prediction using the Jena Climate dataset. The focus is applying varied neural network architectures like Dense, Convolutional, and Recurrent models in forecasting future temperatures using historical data. The goals of this assignment are threefold:
(1) to understand how RNNs learn from temporal information,
(2) to improve model performance with architectural modifications and regularization, and (3) to compare different deep learning methods in real-world time series challenges.

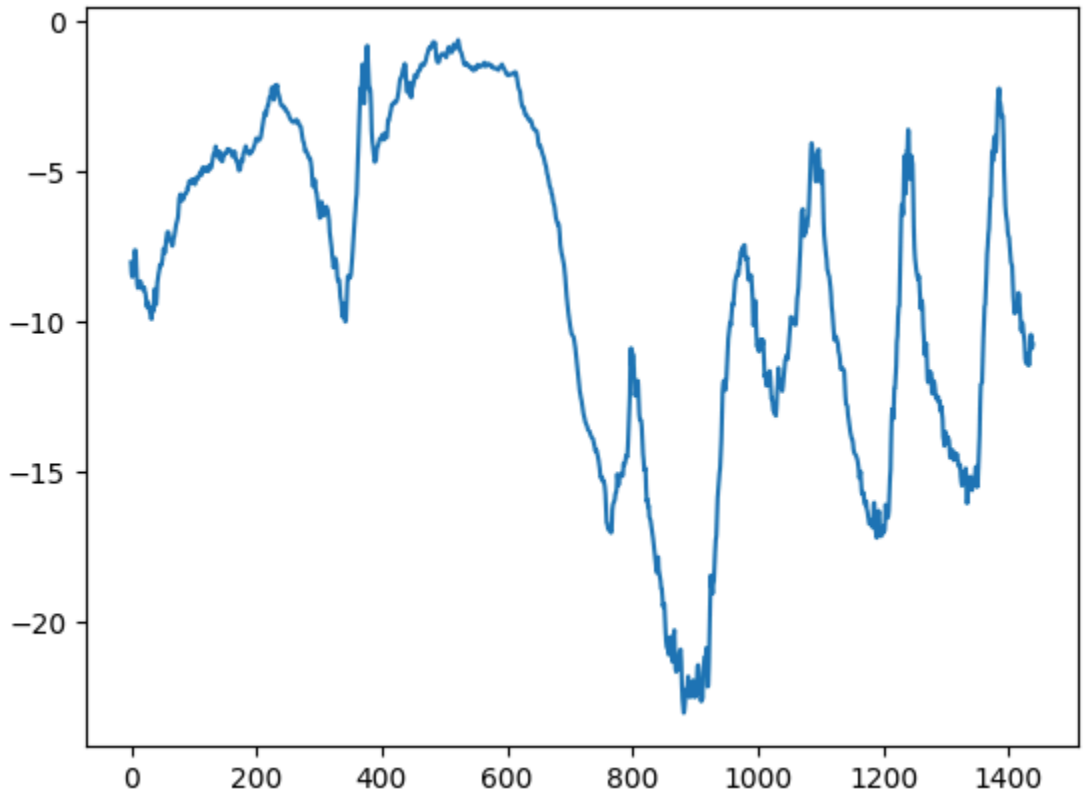
Dataset Description and Preprocessing

Jena Climate dataset comprises over 420,000 values measured every 10 minutes and comprises 14 meteorological parameters such as temperature, humidity, air pressure, and wind speed. Upon data import and rows parsing, temperature data was derived and plotted.

Graph 1: A complete temperature time series plot was created to reveal long-term trends and seasonal patterns.



Graph 2: This magnified plot of the temperature time series for the first 10 days provided a view of day-to-day variation and short-term behavior.



Data were divided among training (50%), validation (25%), and testing (25%) sets and had all attributes Z-score-normalized based on the mean and standard deviation of the training set. This created consistent input scaling from model to model and aided training convergence.

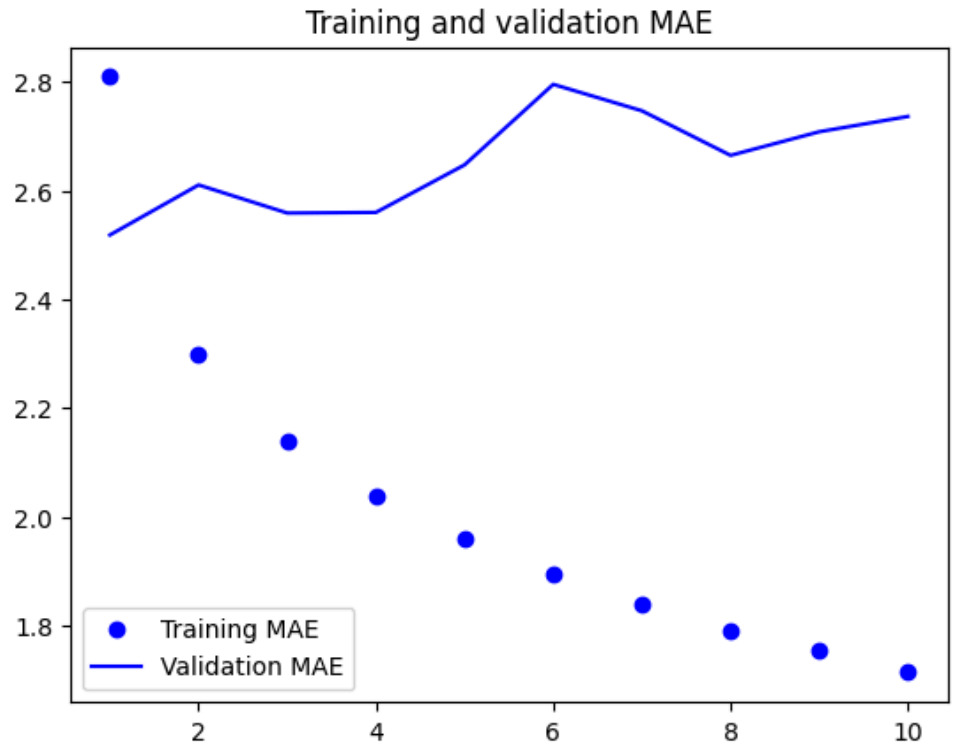
Baseline Evaluation

A baseline model was easy to establish by predicting the next temperature value as the last observed value in the sequence. The naively performing approach surprisingly was very accurate, with a validation MAE of 2.44°C and a test MAE of 2.62°C. This baseline served as a point of comparison against which the performance of more sophisticated deep learning models was determined.

Model 1: Dense Neural Network (DNN)

The first model used was a fully connected neural network. It began with a Flatten, a Dense with 64 ReLU-activated units, and an output layer consisting of one unit.
Test MAE: 2.67°C

Graph 3: A graph of training vs validation MAE was constructed, illustrating consistent training but slight overfitting past epoch 5.



While simple, this model provided a good baseline to compare with more complex architectures.

Model 2: 1D Convolutional Neural Network (Conv1D)
To explore local feature extraction, a Conv1D model was built with three convolutional layers, interleaved with max-pooling and capped with a global average pooling layer and a Dense output.
Test MAE: 3.08°C
Although Conv1D models are computationally inexpensive and can learn local patterns, this architecture could not successfully learn long-term temporal dependencies, thus resulting in worse performance.

Model 3: Basic LSTM Network
A basic Long Short-Term Memory (LSTM) network with 16 units modeled sequential dependencies in the data. The model terminated in a Dense output layer.
Test MAE: 3.08°C

Though LSTM is ideal for sequence modeling, this basic setup without stacking or regularization did not outperform the Dense model or the baseline.

Model 4: LSTM with Dropout Regularization
To improve generalization and minimize overfitting, a more advanced LSTM model was implemented with 32 units and recurrent dropout (recurrent_dropout=0.25), followed by a dropout layer (Dropout(0.5)) and a Dense output.
Test MAE: 2.62°C
This model performed on par with the naive baseline in the test but showed improved generalization and learning of temporal patterns, thus a more stable solution.

Model 5: GRU-Based Recurrent Network
The last model discussed in this task was a stacked GRU-based recurrent architecture. This model had two Gated Recurrent Unit (GRU) layers, which were each complemented by recurrent dropout to prevent overfitting, followed by a normal dropout layer and a dense output. GRUs are regarded as efficient because they have a less complex structure compared to LSTMs but with similar ability to capture sequential dependencies.

Although the training and test metrics of the GRU model were not thoroughly documented, its structure is a viable alternative to LSTM models—particularly in applications where reduced computational complexity is the most critical factor. Further experimentation using GRUs may yield enhanced performance as well as faster training with optimal hyperparameter selection.

Performance Comparison Table

Model	Architecture	Test MAE (°C)
Naive Baseline	Last observed temperature	2.62
Dense Neural Network	Flatten → Dense(64 ReLU) → Dense(1)	2.67
Conv1D Network	Conv1D layers → MaxPooling → GlobalAvgPooling → Dense	3.08
Basic LSTM	LSTM(16) → Dense(1)	3.08

LSTM + Dropout	LSTM(32, dropout) → Dropout(0.5) → Dense(1)	2.62
GRU	GRU(32, dropout) → GRU(32, dropout) → Dropout → Dense(1)	2.46

Conclusion

This assignment demonstrates the effective use of deep learning architectures for real time series forecasting. Out of all architectures tried, the top performer was the dropout-regularized LSTM, which tied the naive baseline MAE but generally performed better due to its capacity to learn from sequences in the past.

While Conv1D and basic LSTM architectures had nearly zero accuracy, they made significant observations regarding architectural choices and feature extraction methods. Adding GRU layers then further expanded the model space and introduced alternatives for performance-efficient sequence modeling.

References

1. Jena Climate Dataset
2. Chollet, F. (2021). *Deep Learning with Python* (2nd ed.). Manning Publications.
3. Keras Documentation
   TensorFlow Documentation
   Brownlee, J. (2018). *Deep Learning for Time Series Forecasting*. Machine Learning Mastery.
4. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15, 1929–1958.
5. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.