BEYKOZ UNIVERSITY

GRADUATE PROGRAMS INSTITUTE

A Comparative Study of Classical and Deep Learning Models for Wind Speed Forecasting

Kimia Sheykh Farshi

MASTER'S THESIS

İstanbul, June 2025

MASTER'S THESIS

Bilgisayar Mühendisliği (İngilizce) Yüksek Lisans Programı

COMPUTER ENGINEERING MASTER'S PROGRAM WITH THESIS

MASTER'S THESIS ADVISOR

Asst. Prof. Dr. İnal Begüm TURNA DEMİREL

MASTER'S THESIS JURY MEMBERS

Asst. Prof. Dr. Gizem TEMELCAN ERGENECOŞAR

Asst. Prof. Dr. Burçin KÜLAHÇIOĞLU

BEYKOZ ÜNİVERSİTESİ

T.C.
BEYKOZ ÜNİVERSİTESİ
LİSANSÜSTÜ PROGRAMLAR ENSTİTÜSÜ MÜDÜRLÜĞÜ

20/06/2025

Yüksek Lisans Tez Onay Belgesi

Enstitümüz Bilgisayar Mühendisliği Anabilim Dalı Bilgisayar Mühendisliği (İngilizce) Tezli Yüksek Lisans Programı 2230140093 numaralı öğrencisi Kimia Sheykh Farshi'nin "A Comparative Study of Classical and Deep Learning Models for Wind Speed Forecasting" adlı tez çalışması Enstitümüz Yönetim Kurulunun ../../…. tarih ve 20../.. sayılı kararıyla oluşturulan jüri tarafından oybirliği ile Tezli Yüksek Lisans tezi olarak kabul edilmiştir.

Öğretim Üyesi Adı Soyadı	**İmzası**

Tez Savunma Tarihi :20/06/2025

1)Tez Danışmanı: Dr. Öğr. Üy. İnal Begüm TURNA DEMİREL

2) Jüri Üyesi : Dr. Öğr. Üy. Gizem TEMELCAN ERGENECOŞAR

3) Jüri Üyesi : Dr. Öğr. Üy. Burçin KÜLAHÇIOĞLU

Not: Öğrencinin Tez savunmasında **Başarılı** olması halinde bu form **imzalanacaktır**. Aksi halde geçersizdir.

CONTENTS

Özet (Türkçe)

Klasik ve Derin Öğrenme Modellerinin Rüzgâr Hızı Tahmini İçin Karşılaştırmalı Bir İncelemesi

Bu çalışmada, zaman serisi modelleri (klasik modeller olan ARMA, ARIMA ile derin öğrenme modelleri LSTM, CNN) rüzgâr hızı tahmini için iki meteorolojik veri kümesine dayalı olarak karşılaştırılmıştır. Literatür taraması, her bir yaklaşımın güçlü ve zayıf yönlerini sunmuş ve veri ölçeği, çözünürlük ile kalitenin tahmin sonuçları üzerindeki etkisini ortaya koymuştur. Nicel sonuçlar, yüksek frekanslı verilerde LSTM'nin, ARIMA'ya kıyasla MSE'yi yaklaşık %99,3 oranında azalttığını; buna karşın ARIMA'nın küçük veri kümeleri ve kısa vadeli tahminler için uygun olduğunu göstermiştir. Ardından, modeller küçük bir Türk veri kümesine uygulanmış ve önerilen eğriler, Almanya'daki geniş kapsamlı Jena veri tabanı ile karşılaştırılarak test edilmiştir. Sonuçlar, klasik modellerin sınırlı verilerle kısa vadeli tahminler için daha uygun olduğunu; derin öğrenme yöntemlerinin ise geleneksel yöntemleri aşmak için yüksek miktarda ve kaliteli verilere ihtiyaç duyduğunu göstermiştir. Bu bulgular, yenilenebilir enerji planlamacılarına veri kümelerinin özellikleri ve operasyonel gereksinimleri doğrultusunda model seçimi konusunda değerli öneriler sunmuştur.

Keywords: Rüzgar hızı tahmini, ARMA, ARIMA, LSTM, CNN

Abstract (English)

A Comparative Study of Classical and Deep Learning Models for Wind Speed Forecasting

This study examined the performance of several time series models (both classical models like ARMA, ARIMA and deep learning models such as LSTM, CNN) are compared in terms of both classical and deep learning models for predicting wind speed based on two meteorological datasets. The literature review presented strengths and weaknesses for each approach, and the influence of data scale, resolution, and quality on the forecasting results. The quantitative result demonstrated that for the high-frequency data LSTM reduces the MSE ~99.3% comparing with ARIMA, when ARIMA is good for short-term prediction and small datasets. Then, the models were used for a small Turkish dataset, and the proposed curves were tested against the large group of Jena database in Germany. Results indicated that while classical models are better suited for forecasting over the short term with limited data, deep learning methods require substantial, high-quality data to surpass traditional forecasting methods. These findings provide valuable recommendations for renewable energy planners on model selection with respect to the characteristics of datasets and their operational requirements.

Keywords: Wind speed forecasting, ARMA, ARIMA, LSTM, CNN

SYMBOLS

t : Current position or the arrival point for Q or q : Quality index

Q_t: Quality set

y_y: Current value at time t

ϕ : AR coefficients

t_0: Beginning of the time interval

T : End of the time interval

θ : MA coefficients

ϵ_t : Error term at time

AR: Autoregressive model; uses past values yt−1,yt−2,…

MA: Moving Average model; uses past errors εt−1,εt−2,…

B : Backshift operator (e.g., Byt=yt−1)

D : Differencing order to achieve stationarity

P : Autoregressive order (e.g., p=5)

Q : Moving Average order (e.g., q=1

ABBREVIATIONS

NWP : Numerical Weather Prediction

ARIMA: AutoRegressive Integrated Moving Average

ANN : Artificial Neural Networks

SVM : Support Vector Machines

ML : Machine Learning

AR :AutoRegressive

MA : Moving Average

MSE : The Mean Squared Error

CNN : Convolutional Neural Networks

LSTM : Long Short-Term Memory

MAE : Mean Absolute Error

MAPE : Mean Absolute Percentage Error

SVR : Support Vector Regression

GANs : Generative adversarial networ

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Forecasting wind speed has become a vital component in both atmospheric science and renewable energy planning, particularly as global reliance on wind power continues to expand. Wind prediction supports optimal turbine performance, grid stability, and efficient energy dispatch, making it highly relevant to industry applications (Liu & Wang, 2019; Wang et al., 2022). With the rising demand for clean energy comes the need for reliable forecasting systems that can cope with diverse temporal and spatial conditions.

In response, the research community has explored a diverse set of modeling strategies. Traditional statistical methods, including Autoregressive Moving Average (ARMA) and Autoregressive Integrated Moving Average (ARIMA), have long served as standard forecasting tools due to their interpretability and effectiveness in short-term predictions (Box et al., 2015; Hyndman & Athanasopoulos, 2018). Nevertheless, their performance diminishes when addressing nonlinear dynamics or multivariate meteorological inputs, particularly in complex, evolving atmospheric conditions (Rehman et al., 2020).

The precise prediction of wind speed is essential for the optimal management of energy and the planning of renewable energy systems. Classical statistical models, such as the ARIMA, are good for capturing linear trends in time series, but they don't do a good job modeling nonlinear relationships. In order to circumvent this limitation, several hybrid models, using both statistical and deep learning techniques, have been suggested. For example, Mohapatra, Patel, Sahoo and Tripathy (2023) provides a hybrid framework by combining ARIMA model with Kalman filter and LSTM networks. It allows for linear and nonlinear dynamics, and outperforms the individual models in forecasting wind speed time series. This shows the promising results of fusing traditional time series methods and state-of-the-art neural network architectures for improved forecasting in the wind energy domain.

Advances in machine learning and artificial intelligence have given rise to new generations of forecasting models, most notably deep learning architectures such as Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs). These models excel at learning complex temporal dependencies and extracting features directly from raw sequential inputs without heavy manual preprocessing (Goodfellow et al., 2016; LeCun et al., 2015; Hochreiter & Schmidhuber, 1997). However, their success depends heavily on access to large, high-quality datasets, which are not always available in real-world scenarios (Zhang-Wiley et al., 2022).

Hybrid and ensemble approaches have emerged as promising alternatives, combining classical and machine learning methods to balance predictive accuracy, interpretability, and computational efficiency (Chen et al., 2023; Kumar & Singh, 2021). In addition, transfer learning using satellite-based and numerical weather prediction (NWP) data offers ways to enhance forecasting in data-scarce environments (Pan & Yang, 2010).

Despite these developments, there remains a clear research gap: few studies have systematically compared classical time series methods with deep learning approaches across datasets of varying scale, resolution, and quality. Most prior work focuses either on small-scale case studies or on large, pre-cleaned datasets, leaving limited guidance on model suitability under real-world constraints.

This research is also timely. Global renewable energy policies and decarbonization strategies are accelerating the deployment of wind power, with ambitious targets for 2030 and beyond. Accurate wind forecasting not only improves operational efficiency but also reduces financial risks in energy markets and supports the stability of increasingly renewable-driven power grids. In this context, evaluating the conditions under which different forecasting models succeed or fail is of direct practical importance.

This chapter provides a critical review of existing literature, focusing on both classical time series models (ARMA, ARIMA) and deep learning models (LSTM, CNN) in the context of wind speed forecasting. It assesses model appropriateness in terms of data scale, resolution, and application setting, while highlighting methodological trade-offs such as interpretability versus

computational cost and generalizability. More specifically, it outlines recent empirical evidence comparing these models in different contexts (Li et al., 2020; Zhang et al., …; Ehsan et al., 2020). In doing so, this review identifies shortcomings in current research and establishes the theoretical basis for the comparative modeling strategies applied in the following chapters.

Chapter 2: Literature Review

Forecasting wind speed is a critical task in renewable energy planning and atmospheric science. Reliable forecasts improve the integration of wind power into energy grids and enable proactive resource management. Over the past decades, a variety of modeling approaches have been proposed ranging from classical statistical techniques like Autoregressive Moving Average (ARMA) and Autoregressive Integrated Moving Average (ARIMA) to modern deep learning models such as Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs). This chapter critically reviews the literature on these models, their strengths and weaknesses, and how their performance is affected by data availability and structure (Li et al., 2020; Zhang et al., 2021).

Wind speed prediction is of critical importance in both maximizing the output of a turbine and in reducing the uncertainty of wind energy generation. The prediction horizon can be divided into very short-term (seconds to minutes), short-term (minutes to hours), medium-term (hours to days) and long-term (days to weeks) (Liu & Wang, 2019). Predictions show that accuracy is determined by a temporal resolution of data, site spatial structure, and nature of the meteorological parameters used in the input model (Wang et al., 2022).

New studies underline the interest in hybrid models integrating statistical and machine learning approaches due to complementary aspects. Moreover, extensive attention has been paid to ensemble methods and transfer learning, especially for those places lacking historical materials. The existence of high-frequency data, satellite-based meteorological variables, and numerical weather prediction (NWP) models has also motivated the construction of data-rich forecasting systems (Chen et al., 2023; Kumar & Singh, 2021).

Alongside these, researchers have also explored decomposition–reconstruction models that integrate statistical and deep learning components (e.g., ICEEMDAN + MFE + LSTM + INFORMER). For example, Wang, Shen, Ai, and Li (2023) proposed a hybrid model combining ICEEMDAN, MFE, LSTM, and

INFORMER, achieving significantly higher accuracy than seven alternative models across multiple metrics.

2.1 Classical Time Series Models: ARMA and ARIMA

Statistical forecasting models have long been applied to univariate meteorological data due to their interpretability and efficiency. The ARMA model, suitable for stationary time series, combines autoregressive (AR) and moving average (MA) components to capture temporal dependencies. When data exhibit trends or non-stationarity, the ARIMA model introduces differencing (the "I" component) to stabilize the mean over time (Box et al., 2015).

Empirical studies have validated the utility of ARIMA in wind speed prediction, particularly in short-term scenarios. For instance, Alsamamra et al. (2024) applied ARIMA to wind speed data in Palestine and achieved satisfactory results for short-term forecasts. Similarly, Yıldız and Özdemir (2023) compared LSTM and ARIMA in Turkey's Gelibolu region, finding that ARIMA performed competitively in data-constrained settings.

Despite their simplicity, ARMA and ARIMA models require rigorous preprocessing, including stationarity tests (e.g., Augmented Dickey-Fuller) and autocorrelation analysis (e.g., ACF, PACF), to ensure robustness. Their parameter selection can be optimized using techniques such as grid search and information criteria like AIC or BIC (Hyndman & Athanasopoulos, 2018).

However, they are limited in representing nonlinearities and nontrivial interactions among meteorological variables. They can fall short when it comes to long-range predictions or highly seasonal and volatile patterns. Additionally, they are restricted to linear relationships, and have difficulty dealing with multivariable exogenous inputs (e.g., temperature, pressure, humidity) for better accuracy. To tackle some of these issues, extensions such as SARIMA (Seasonal ARIMA) and ARIMAX (ARIMA with exogenous variables) have been proposed with improved handling of seasonality and multivariate settings (Rehman et al., 2020). Despite these improvements, conventional models still rely highly on elaborate feature engineering and expert-designed data manipulation, which can be time-consuming and subjective.

2.2 Deep Learning Models: LSTM and CNN

Time series forecasting has been revolutionized by deep learning models, particularly Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs), which can capture complex nonlinear temporal dependencies (Goodfellow et al., 2016). The memory cell structure and gating mechanism of LSTMs are very well suited for learning long-range dependencies by resolving the vanishing gradient problem of vanilla RNNs. This allows models to remember information from long-term input sequences despite fading memory, which suits modeling slow wind speed changes over time (Hochreiter & Schmidhuber, 1997). CNNs, in contrast, can identify short-range and periodic patterns with convolutional layers, distilling salient features from raw input sequences without manual feature extraction (LeCun et al., 2015). When used for time series data, CNNs can capture local temporal characteristics and scale across spatial resolutions, particularly when pooling and dilation are applied (Bai et al., 2018).

These models have been used in several recent studies analyzing meteorological datasets at large scales. Elsaraiti and Merabet (2021) as well as Ehsan et al. (2020) found that LSTMs outperform traditional models for long-term wind forecasts when trained on high-resolution data. These models also show good ability to learn temporal dynamics, particularly when exogenous variables like temperature, humidity, and barometric pressure are included. Multivariate LSTM models have been proposed to improve accuracy by incorporating spatio-temporal dependencies from neighboring stations (Zhang et al., 2022).

However, recent comparative evidence also shows that carefully designed hybrids (decomposition + DL/transformer backbones) can outperform plain deep networks, especially by denoising and isolating nonstationary components prior to learning.

Nevertheless, Zhang-Wiley et al. (2022) highlighted significant challenges in applying these models to small datasets, where their complexity can cause overfitting and poor generalization. To mitigate these issues, techniques such as dropout regularization, early stopping, and data augmentation have been adopted. Transfer learning has also gained attention as a method to improve generalization

by pretraining models on large related datasets before fine-tuning on smaller site-specific data (Pan & Yang, 2010). Model interpretability remains a concern, as deep learning models function as black boxes, making it difficult to trace the reasoning behind predictions. Attention mechanisms and explainable AI (XAI) tools are being explored to improve transparency (Ribeiro et al., 2016).

Hybrid architectures combining LSTM and CNN (e.g., CNN-LSTM and ConvLSTM) leverage the strengths of both models. These hybrids have demonstrated effectiveness in capturing spatiotemporal information jointly and improving forecast performance, especially in complex or noisy meteorological conditions (Shi et al., 2015). However, deep learning models require more computational resources and longer training times, and their performance depends heavily on proper hyperparameter tuning, input window size, and sequence length, posing practical difficulties for real-time wind forecasting (Karim et al., 2019).

2.3 Data Constraints and Model Suitability

The suitability of forecasting models is closely tied to dataset characteristics. Classical models like ARMA/ARIMA perform well on small and moderately complex datasets, offering transparent and stable forecasts (Hyndman & Athanasopoulos, 2018). In contrast, LSTM and CNN require extensive historical data and dense sampling intervals to leverage their full potential (Zhang et al., 2021).

The Turkish dataset used in this study, comprising 5,384 hourly records over five months, exemplifies a small-scale, irregularly sampled dataset. Such data are insufficient for training deep networks, which tend to overfit limited observation windows. Consequently, ARIMA and ARMA models remain more appropriate, as shown by their reliable performance on both hourly and daily wind speed series (Alsamamra et al., 2024).

Conversely, the Jena dataset from Germany provides over 175,000 entries at 10-minute intervals across multiple years. This data richness enables effective deep learning applications, with LSTM and CNN models outperforming ARIMA, especially for complex and long-term trends (Elsaraiti & Merabet, 2021).

2.4 Comparative Studies and Mixed Results

While some literature suggests deep learning models outperform classical techniques, the advantage often depends on dataset scale and model tuning. For example, in this thesis, ARIMA slightly outperformed ARMA in both the Jena and Turkish datasets, particularly after differencing to manage non-stationarity. However, the margin was narrow, and in short-term forecasting, both models yielded similar accuracy levels (Yıldız & Özdemir, 2023).

At the same time ,while many papers highlight the advantages of deep learning, recent hybrid approaches from 2023–2024 have demonstrated consistent, peer-reviewed gains over single-family models (both classical and DL), suggesting a promising direction for future work and real-time deployment. For example, Wang et al. (2023) demonstrated superior performance with their ICEEMDAN–MFE–LSTM–INFORMER hybrid model across multiple forecast horizons, supporting hybridization as a promising direction (Wang et al., 2023).

SARIMA models have been proposed as an extension to ARIMA for handling seasonality, and recent studies (e.g., Tyass et al., 2022) demonstrate their value in modeling diurnal and weekly cycles in wind speed. Nonetheless, these models still face limitations when applied to short time spans, such as the Turkish dataset (Rehman et al., 2020).

2.5 Limitations of Deep Learning in Small Datasets

Deep learning models like CNNs and LSTMs show excellent performance when trained on large-scale, high-quality datasets across various domains (Goodfellow et al., 2016). Nevertheless, when applied to small-scale or limited datasets, inherent challenges must be acknowledged (Zhang-Wiley et al., 2022).

Data insufficiency for capturing complex patterns: Deep learning models require large, diverse datasets to learn complex patterns effectively. Small datasets lack sufficient variability, impeding meaningful learning of long-range temporal or spatial dependencies, resulting in poor predictions (Bai et al., 2018).

Overfitting risks due to high model capacity: The complexity and large parameter count in deep networks lead to overfitting on limited data, where the model memorizes training samples but fails to generalize to new data (Goodfellow et al., 2016).

Training instability in resource-constrained settings: Training deep networks demands extensive computational resources and careful hyperparameter tuning. In small data regimes, optimization noise can cause unstable training dynamics like oscillations or premature convergence, and regularization methods like dropout may be less effective (Karim et al., 2019).

Lack of interpretability limiting scientific insight and operational trust: Deep learning models operate as black boxes, which becomes problematic with small datasets where performance and behavior are harder to evaluate. Interpretability is crucial for scientific and practical applications to build trust and extract domain knowledge (Ribeiro et al., 2016).

Though CNN and LSTM models are powerful in time series forecasting, their application to small-scale data should be approached carefully. Traditional statistical or hybrid models often provide more robust and accurate results under data constraints. Future work should explore data augmentation, transfer learning, and model simplification to address these challenges (Pan & Yang, 2010; Rehman et al., 2020).These considerations are particularly relevant because deep models are inherently data-hungry and prone to overfitting when datasets are short or have limited seasonal coverage.

On a one-month offshore dataset, a peer-reviewed comparison reported that SARIMA outperformed LSTM and GRU at multiple heights; e.g., at 25 m height, SARIMA achieved(RMSE = 0.8486)and (MAE = 0.6168) versus (LSTM's RMSE = 0.9695) and (MAE = 0.7175)about 12.5% lower RMSE and 14% lower MAE attributed to overfitting and insufficient seasonal coverage for the deep models. Regularization, early stopping, and transfer learning help, but when data are short and noisy, classical or hybrid models can be more robust and computationally efficient, a practical consideration for real-time wind operations and sites with limited historical data.

2.6 Summary and Research Gap

Existing literature highlights the strength of classical time series models in short-term, data-limited forecasting, especially when preprocessing and parameter tuning are applied (Hyndman & Athanasopoulos, 2018). Deep learning, though powerful, is data-hungry and computationally demanding. This contrast is

particularly apparent in contexts where data quality or computational resources are limited, emphasizing the continued relevance of classical approaches for practical applications.

At the same time, literature highlights a clear split: classical models offer robustness and interpretability for small or short-term settings, whereas deep models and especially recent hybrid models excel on large, high-resolution datasets. The gap addressed in this thesis is a principled, side-by-side evaluation across data regimes, both with and without modern hybridization, aiming to clarify when each approach is preferable for operational wind forecasting.

Many studies focus on large-scale deep learning deployments or controlled benchmarks, leaving gaps in understanding their applicability to real-world, small-scale meteorological datasets. Limited research compares model performance under different noise, resolution, and seasonality conditions.

This thesis contributes by comparing ARMA, ARIMA, LSTM, and CNN across datasets of varying size and resolution, identifying conditions where classical models remain preferable. It provides practical insights on robustness, accuracy, and computational needs, aiding practitioners working

Chapter 3 : Methodology

In this project, we wanted to predict wind speed using two kinds of models. The first type is classical models like ARMA and ARIMA, which are good for simple patterns in data. The second type is deep learning models like LSTM and CNN, which can find more complex patterns. We worked with two different datasets. One dataset comes from a university in Turkey. It has short-term data, covering only a few months. The second dataset is from the Max Planck Institute in Jena, Germany and has many years of data (Kaggle, n.d.; Utku, Can, & Aksoy, 2023).

We trained and tested the ARMA and ARIMA models on the Turkish dataset and the deep learning models (LSTM and CNN) on the Jena dataset. Then, we compared the results to see which models worked better on each dataset. We also checked how the size and quality of data affect model performance. Our goal was to understand when classical models are enough and when we need deep learning.

This study evaluates classical and deep learning forecasting models on two meteorological wind speed datasets: a short-term Turkish university dataset and a long-term dataset from the Max Planck Institute for Biogeochemistry (MPI-BGC) in Jena, Germany. The primary objectives are:

First, To compare ARMA/ARIMA models with LSTM/CNN architectures using the high-quality Jena dataset.

Second, To assess classical models' limitations when applied to the small-scale Turkish dataset.

Third, To explore cross-dataset insights by comparing Turkish ARIMA forecasts with Jena deep learning predictions, emphasizing the effects of data size and model suitability.

3.1 Dataset Description

We used two datasets in this project to forecast wind speed. The first dataset is from a meteorological sensor at a university in Turkey. It includes wind speed values recorded every hour from July 4, 2023, to February 9, 2024. In total, there

are 5,384 data points. Each record has a timestamp and a wind speed value. The wind speeds were recorded as text (for example, "5.3"), and missing values were written as "--.-". To use this data, we had to clean it. We converted the timestamps to real date-time format, removed or fixed the missing values, and changed the wind speeds to numbers (Meteorological Dataset in Turkey, 2023).

The second dataset is from the Max Planck Institute for Biogeochemistry (MPI-BGC) in Jena, Germany. This dataset is much larger and covers many years of weather data, recorded every 10 minutes. It includes wind speed, temperature, and other weather features. We used only the wind speed values in this project. Since the Jena dataset is bigger and cleaner, we used it with deep learning models to learn complex patterns over a longer time period.

Turkish University Wind Speed Dataset

The dataset used in this study was obtained from a meteorological sensor located in Turkey and reflects typical monitoring practices in the region, similar to those described by Akyüz and Yılmaz (2019). It consists of hourly wind speed records collected between July 4, 2023, and February 9, 2024, totaling 5,384 samples. The dataset contains two main variables: a timestamp column, which required datetime conversion for analysis, and wind speed values measured in kilometers per hour. These wind speed values were initially recorded as strings, with some entries marked by “--.-” to indicate missing values.

Several challenges emerged during preprocessing. The most significant issues were the presence of missing values, which required careful imputation, and non-numeric entries that necessitated type conversion before any modeling could be performed. These preprocessing steps were essential to ensure the dataset's suitability for statistical and machine learning analyses.

Software and Tools Used in the Project

To prepare, analyze, and model the wind speed data, we used several Python-based tools and libraries that supported both classical time series modeling and deep learning approaches.

Pandas was central to our data manipulation tasks. It allowed us to load the raw CSV files, clean and structure the data, and perform operations such as removing or imputing missing values and formatting the time variable. Commonly

used functions included read_csv(), to_datetime(), and groupby() (Sundaram, 2024), all of which were instrumental in converting the raw data into a usable time series format.

NumPy supported mathematical operations and array handling. It facilitated numerical computations and helped prepare the data for model input by enabling efficient manipulation of arrays and matrices, which are core components in time series modeling.

For classical time series modeling, we employed the *Statsmodels* library to build and evaluate ARMA and ARIMA models, which are well-established approaches for analyzing temporal dependencies in wind speed data. Model selection was performed by systematically varying the autoregressive (p) and moving average (q) parameters within predefined ranges and comparing performance using mean absolute error (MAE). This grid search approach provided a practical alternative to relying solely on autocorrelation-based diagnostics. The methodological workflow aligns with recommended practices for statistical time series modeling as outlined in Brownlee (2023) and the official *Statsmodels* documentation (Statsmodels, n.d.).

To evaluate model performance, we relied on Scikit-learn (sklearn). Specifically, we used the mean_absolute_error function to quantify the difference between predicted and actual wind speed values. This metric provided a straightforward and interpretable measure of prediction accuracy.

For deep learning experiments, we used TensorFlow and Keras, which offered robust tools for designing and training neural networks. Keras, running on top of TensorFlow, allowed us to build and experiment with architectures such as CNN and LSTM. Key components included Conv1D, LSTM, Dense, and Dropout layers, all of which contributed to the model's ability to learn from temporal features in the data.

Core Libraries for Data Handling & Modeling

The core Python libraries used throughout this project provided critical functionality for managing and modeling the wind speed dataset. Pandas played a central role in data loading, cleaning, and manipulation. Functions such as pd.read_csv(), groupby(), and to_datetime() were employed to structure and convert

the dataset into an analysis-ready form. NumPy supported these efforts by enabling fast and efficient numerical operations, especially when working with arrays during preprocessing and transformation.

Together, these libraries provided a structured pipeline: Pandas and NumPy handled data preparation, Statsmodels enabled classical time series modeling, and Scikit-learn supported model validation and comparison. This modular approach allowed for flexibility and transparency throughout the analysis workflow.

Preprocessing Steps

Prior to modeling, the dataset required careful preprocessing to ensure data quality and usability. The wind speed variable, initially recorded as strings, contained some non-numeric values that could not be used in quantitative analysis. These invalid entries were handled by coercing errors during type conversion and then removing any resulting null values. This step was essential to maintain the integrity of the dataset.

Next, the timestamp column was transformed into a proper datetime format using Pandas' to_datetime() function. This conversion enabled temporal indexing and allowed the data to be grouped into various time intervals. Specifically, the data was resampled into hourly, daily, and monthly periods, and for each time group, the average, maximum, and minimum wind speeds were calculated. These aggregations offered different perspectives on temporal trends and improved the dataset's utility for time series forecasting.

All of these preprocessing and transformation steps such as type conversion, datetime formatting, and statistical aggregation were carried out using Pandas and NumPy, adhering to data cleaning practices outlined by Sundaram (2024). This process ensured that the dataset was clean, well-structured, and appropriate for both statistical and machine learning-based modeling.

Key Observations from the Data

The wind speed dataset begins in early July 2023, with data points recorded at hourly intervals. For model evaluation, the dataset was partitioned into training and test subsets. Specifically, data from November 6, 2023, onward was reserved for testing in hourly models, while data from October 30, 2023, was set aside for daily forecasts. Initial exploration of the dataset revealed notable fluctuations in

wind speed. Hourly values often varied significantly at times rising sharply from 0 km/h to 17 km/h within a short period. This variability reflects the dynamic nature of wind behavior in the region.

At the daily level, wind speed patterns exhibited alternating periods of calm and turbulence. Some days maintained consistently low speeds around 3 km/h, while others experienced more pronounced spikes. On a monthly scale, a clear seasonal trend emerged. Wind speeds in July and August averaged between 6 to 9 km/h, indicating stronger and more consistent airflow. In contrast, September displayed calmer conditions, with wind speeds frequently dropping to approximately 3 km/h. These temporal patterns suggest that seasonal and diurnal cycles are important factors in wind behavior, and they influenced how the models were structured and evaluated.

Visualization and Workflow

To better understand these patterns, both static and interactive visualizations were employed. Libraries such as Matplotlib and Seaborn were used to generate standard plots for temporal and statistical analysis, while Plotly Express enabled optional interactive visualizations, offering a more dynamic exploration of fluctuations across different time frames.

The raw data initially presented several structural challenges. The time variable was stored as an object or string in the format DD.MM.YYYY HH:MM:SS, requiring conversion to proper datetime objects. The wind speed values, originally stored as strings (and in some cases non-numeric), were converted into float values for numerical analysis. This preprocessing stage was followed by resampling the data into hourly, daily, and monthly formats, which provided more stable inputs for forecasting models.

The modeling phase began with the construction and evaluation of ARIMA models. A grid search was conducted to test multiple combinations of the (p, d, q) parameters, and each configuration was validated using a walk-forward approach. The mean absolute error (MAE) metric was employed to quantify predictive accuracy. Model results were then visualized alongside actual wind speed values to assess fit and reliability.

Technical Strengths and Future Potential

One of the key strengths of this workflow lies in its efficiency and flexibility. The use of Python's scientific ecosystem particularly Pandas, Statsmodels, and Scikit-learn allowed for a lightweight yet powerful analysis pipeline. Furthermore, the modular structure of the codebase ensures scalability, allowing for future adaptation to larger datasets. If necessary, parallel processing frameworks such as Dask or Apache Spark can be incorporated in future work to handle increased data volume or real-time forecasting demands.

Potential Upgrades

This project used simple but powerful tools in Python to handle and analyze wind speed data. It ran well on a small dataset and could easily be scaled up in the future using tools like Dask or Apache Spark. The code was clean and easy to follow. For future upgrades, we could use models that understand seasonal patterns, like SARIMA, or try advanced models like Prophet and LSTM that are better for complex changes in data over time. These upgrades could make the predictions even more accurate. In a recent study, (Aziz et al. (2023)) compared SARIMA and Prophet for forecasting solar and wind resources in remote areas of Australia. They found that Prophet outperformed SARIMA in both RMSE and MAE, making it a strong candidate for time series forecasting in energy applications.

SARIMA (seasonality), Prophet/LSTM (complex patterns).

MPI-BGC Jena Meteorological Dataset

The Jena Wind Speed Dataset was sourced from the Max Planck Institute for Biogeochemistry in Jena, Germany. It spans a comprehensive period from 2004 to 2023, consisting of more than 175,000 records sampled at 10-minute intervals. The primary variable of interest in this study is wind speed (represented as wv (m/s)), although other meteorological variables such as temperature, humidity, and pressure are also included. These additional variables, while useful, were not modeled in this research. The dataset's main advantages include its pre-cleaned format, high reliability, and minimal missing data, making it particularly well-suited for training deep learning models that require consistent, structured input.

To handle and analyze the dataset, a variety of Python libraries were employed. Pandas was used for loading the CSV files, managing time-based

indexing, and performing aggregation operations. NumPy supported numerical calculations and efficient array handling, while Matplotlib was leveraged for visualization purposes. These visualizations included time-series plots to observe long-term trends, histograms for distribution analysis, and box plots to detect outliers and anomalies in the wind speed data.

Before the modeling phase, the data underwent essential preprocessing. Initially, raw data was downloaded from online ZIP files covering the years 2004 to 2020. The cleaning process involved removing invalid entries such as negative wind speeds or placeholder values like –9999.0, which are common indicators of sensor failure or transmission errors. Linear interpolation was used to fill in small gaps and maintain continuity in the time series. After cleaning, the data was resampled to 10-minute intervals to standardize the frequency, and statistical measures like mean, minimum, and maximum wind speeds were computed to characterize its temporal structure.

These preprocessing steps closely follow the standard methodologies outlined in prior work involving the Jena dataset, such as those referenced in Keras (n.d.). Visual inspection through time-series aggregations helped reveal patterns on hourly and daily scales, while 2D histograms provided insight into relationships between variables. Box plots were especially useful for highlighting unusual values that could distort model learning. This multi-stage processing workflow ensured that the dataset was well-prepared for both classical and deep learning modeling approaches.

Potential Applications and Dataset Strengths

The Jena wind speed dataset holds significant potential for various scientific and industrial applications. One of the primary areas is weather forecasting, where machine learning models particularly deep learning approaches such as Long Short-Term Memory (LSTM) networks can leverage the high-resolution, time-series data to improve short-term and long-term predictions. In addition, the dataset is valuable for climate trend analysis, enabling researchers to detect subtle patterns and shifts in wind behavior over the course of nearly two decades. Its extensive time range and consistency also make it well-suited for renewable energy studies, especially in

evaluating the feasibility and optimization of wind and solar power generation across different seasons and years.

What sets this dataset apart is its high spatial and temporal resolution, offering wind speed measurements at 10-minute intervals a level of detail that supports fine-grained modeling. Moreover, the dataset's long-term coverage, spanning 17 years, provides a robust basis for analyzing seasonal, annual, and decadal variability. Another notable strength lies in its comprehensive preprocessing; the data has undergone systematic cleaning and interpolation to handle missing or erroneous values, thereby ensuring reliability for computational modeling. These combined features make the dataset an exceptional resource for data-driven environmental research and engineering applications.

3.2 Data Preprocessing

Before using the data for forecasting, we cleaned and prepared both datasets. For the Turkish dataset, the time column was reformatted, missing wind speed values were corrected through linear interpolation, and wind speed was converted from text to numerical values. The data were then grouped into hourly averages to facilitate analysis. For the Jena dataset, the primary variable selected was wind velocity (wv), measured in meters per second. To reduce computational cost and allow rapid experimentation, the dataset was subsampled to the first 500 time steps. This choice provided a practical compromise: while processing the full dataset would have required significant GPU memory and longer training times, the smaller subset enabled efficient evaluation, model comparison, and hyperparameter tuning. Despite the reduction, the subset still captured meaningful temporal patterns and supported a systematic, iterative workflow prior to scaling to the complete dataset. The main preprocessing steps and dataset characteristics are summarized in Table 3.1, which highlights differences such as missing data handling, temporal resolution, train-test splitting, and normalization methods between the Turkish and Jena datasets.

Turkish Dataset

The preprocessing of the Turkish wind speed dataset began with a systematic cleaning phase. The Time column, originally stored in string format, was converted into Python datetime objects using the pandas.to_datetime() function and standardized into three different temporal resolutions, hourly, daily, and monthly by converting the datetime values into Period objects (period[h], period[D], and period[M]). This step ensured that the dataset could be analyzed consistently across multiple time scales. Wind speed entries occasionally contained invalid strings, such as "--.-", which were identified as non-numeric. These invalid records were coerced into missing values and subsequently discarded, leaving only valid numerical observations. The remaining wind speed values were then cast into floating-point numbers to enable statistical computation and time-series modeling.

For analysis, the above filtered values were aggregated over multiple timescales, creating redundant views of wind. In detail, the mean, maximum and minimum wind speeds were calculated in hourly, daily and monthly ranges. The hourly aggregation captured the short-term variability of the wind speeds and was therefore particularly suitable for high resolution predictions and model fitting. The daily aggregation filtered noise by averaging out intra-day variability, while retaining daily rhythms and trends of interest for short-term energy planning. Monthly aggregation offered a perspective along a longer time scale, which can provide seasonally and climatologically relevant information, important for the operation of energy systems and the management of energy resources and capacities , However, since the dataset covered only a Eight-month period, monthly data were insufficient to reliably capture or model seasonal patterns, and therefore seasonality analysis was primarily limited to the hourly and daily scales. By organising the dataset across several temporal granularity levels, the preprocessing pipeline not only cleaned and aligned the data, but also made possible a thorough comparison of ARMA and ARIMA models across various forecast horizons.

Jena Dataset

In preparing the Jena dataset, the primary variable selected for analysis was wind velocity (wv), measured in meters per second. For initial experimentation, the dataset was subsampled to the first 500 time steps. This decision was motivated by

both computational and experimental considerations: processing the full dataset would have required excessive GPU memory and longer training times, while using a smaller subset allowed rapid testing, model comparison, and hyperparameter tuning. By focusing on 500 samples, the models could be evaluated efficiently, meaningful temporal patterns could still be captured, and the experimental workflow remained systematic and iterative, providing a practical compromise before scaling to the full dataset.

The dataset was then split into training and testing sets, using a 66% to 34% ratio. This division was implemented using walk-forward validation, a method well-suited for time series tasks because it preserves temporal order and mimics real forecasting scenarios. Before model training, the wind speed values were normalized to the [0, 1] range using the MinMaxScaler. This step is particularly crucial in deep learning to ensure faster convergence and to prevent the model from becoming biased toward features with larger numeric scales.

3.3 Justification of Preprocessing

This table summarizes the main characteristics and preprocessing steps for the two datasets. It includes how missing data were handled (linear interpolation for Turkish, none for Jena), their temporal resolution (hourly vs. 10-minute), train-test splitting methods (time-based 70-30 for Turkish, randomized 66-34 for Jena), and whether normalization was applied (none for Turkish ARIMA models, MinMaxScaler for Jena LSTM/CNN models).

for Rationale:

The Turkish dataset’s small size favors classical models.

Jena dataset’s large volume and quality enable deep learning.

Table 3.1 Comparison of Turkish and Jena wind speed datasets

Feature	Turkish Dataset	Jena Dataset
Missing Data	Linear interpolation	None (pre-cleaned)
Temporal Resolution	Hourly (resampled from hourly)	10-minute (retained)
Train-Test Split	70-30 time-based	66-34 randomized
Normalization	Not applied (ARIMA handles raw)	MinMaxScaler for LSTM/CNN

Case Study Workflow

In this project, we followed the same steps for each case study. First, we cleaned the data and changed the time format. Then, we split the data into training and testing parts. After that, we built the models ARIMA for classical forecasting, and LSTM or CNN for deep learning when possible. We used tools to check patterns in the data, like ACF and PACF, to choose the best ARIMA settings. Finally, we tested all models using different scores like Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE) to see which one worked best.

Case1: Jena Dataset (ARIMA vs. LSTM/CNN)

This case study demonstrates the application of both statistical and deep learning models on a test scenario involving the Jena climate dataset which is made up of many years worth of high-resolution observations about wind speed. First, we trained an ARIMA model with optimal parameters chosen using ACF and PACF plots. We also compared this with models using similar LSTMs and CNNs

(Convolutional Neural Networks) as those mentioned above. We used 2-layer LSTM architecture which was trained for 50 epochs. We went on to evaluate all three methods using post-model training standard error metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE).The results indicated that deep learning models, especially the LSTM, outperformed the ARIMA model in predictive accuracy. This performance advantage can be largely attributed to the Jena dataset's size, resolution, and low level of missing data conditions under which deep learning models excel. However, this improvement came at the cost of increased computational demand and decreased model interpretability. Therefore, while neural networks are more effective for highly detailed, long-term datasets like Jena's, statistical models such as ARIMA remain preferable in scenarios where transparency and computational efficiency are prioritized over raw predictive performance.

Case Study 2: Turkish Dataset (ARIMA Only)

The Turkish wind speed dataset used in this study was relatively small and contained some missing or incorrect values. To ensure data quality, the dataset underwent a thorough cleaning process before modeling. Given the limited size and sparsity of the data, deep learning models such as LSTM and CNN were excluded from consideration, as these architectures generally require large amounts of training data to perform effectively and avoid overfitting. Instead, an ARIMA model with a differencing order of one (d = 1) was chosen to stabilize the series and address non-stationarity issues. The model was then trained and tested on the cleaned data, yielding fair forecasting results. While the performance was not as strong as that observed with larger datasets such as the Jena dataset, these results highlight ARIMA's suitability for smaller datasets where deep learning methods struggle.

This chapter systematically evaluated classical time series models, particularly ARMA and ARIMA, on the limited Turkish wind speed data, which spanned a short time period and had relatively few observations. The analysis showed that these traditional statistical models are well-suited for short-term

forecasting at hourly and daily levels, achieving reasonable accuracy despite the data constraints. Notably, ARIMA models incorporating differencing consistently outperformed ARMA models by better capturing subtle temporal trends, especially in the hourly time series. However, due to the dataset's short temporal span and lack of seasonal patterns, monthly forecasting was unreliable, demonstrating the limitations of applying these models beyond short-term horizons.

In addition, the chapter critically discussed why deep learning approaches like CNNs and LSTMs are not appropriate for this dataset. These methods rely heavily on large, diverse datasets and complex model structures to generalize effectively. In small, sparse datasets like the one analyzed here, they tend to overfit and yield unstable predictions. Moreover, their complexity reduces interpretability, which is an important consideration for practical applications.

Overall, the findings emphasize the strengths of classical time series models when working with constrained renewable energy datasets. By leveraging simpler, more interpretable, and data-efficient forecasting techniques such as ARIMA, practitioners can produce robust short-term wind speed predictions without the computational overhead and risks associated with deep learning. This understanding is crucial for guiding future modeling decisions and highlights the need to align forecasting methods with the scale and quality of available data, especially in real-world contexts where reliable long-term prediction is essential but data resources are limited.

Case Study 3: Cross-Dataset Comparison

This study compared ARIMA's performance on a small Turkish wind speed dataset with LSTM and CNN models applied to a larger Jena dataset, highlighting how data size and quality influence model choice. ARIMA proved effective and easier to use with limited data, while deep learning models required larger datasets to outperform classical methods. This comparison emphasizes that model selection depends largely on the amount and nature of available data.

The Turkish dataset, with its limited duration and observations, was well-suited to classical time series models like ARMA and ARIMA for short-term (hourly and daily) forecasting. ARIMA consistently outperformed ARMA by better

handling non-stationarity and subtle temporal patterns. However, monthly forecasts were unreliable due to the dataset's constraints.

Deep learning models such as CNN and LSTM, which depend on extensive data and complex tuning, struggled with this limited dataset. Issues like overfitting and poor generalization reduced their practicality for small-scale wind speed prediction.

Overall, the findings suggest that simpler, interpretable models like ARIMA are more suitable for short-term forecasting when data is limited, while deep learning excels only with larger datasets. Aligning model choice with data scale and quality is crucial for reliable wind speed prediction in real-world scenarios.

3.4 Forecasting Models

We used two kinds of models to predict wind speed. The first type was classical models like ARIMA, which use patterns from past wind speeds and past errors. These models are good for small or simple datasets. The second type was deep learning models like LSTM and CNN. LSTM is a smart model that can learn from long sequences of data. CNN works well with patterns that repeat, like daily or weekly trends. We used these deep models only for the Jena dataset because it had enough data. Many studies demonstrate the efficiency of this approach. For instance, (Elsaraiti and Merabet (2021)) directly compare ARIMA and LSTM models for wind speed forecasting, concluding that LSTM delivers superior performance in terms of prediction accuracy. Similarly, (Ehsan et al. (2020)) report that LSTM achieves around 97.8% accuracy in wind speed prediction, outperforming several other statistical and neural network baselines .

Classical Time Series Models

Classical models like ARMA and ARIMA are used when we want to predict values based on past data. AR looks at past wind speed values, while MA looks at past prediction mistakes. ARIMA adds one more step, called differencing, which helps when the data isn't stable. These models are easy to use, need fewer

resources, and work well with smaller datasets like the Turkish one. They are a good choice when deep learning is not possible.

The ARMA model is an established statistical method applied to time series prediction. It's a combination of two components; the autoregressive (AR) part, which depends on the past and the moving average (MA) part that extends the past errors. All of them and in combination make ARMA suitable for detecting linear dependencies in time series.

ARMA (AutoRegressive Moving Average)

Equation:

$$yt = c + \sum_{i=1}^{p} \phi i\, yt - i + \sum_{j+1}^{q} \theta j \epsilon t - j + \epsilon t$$

Here, yt represents the current value of the series at time ttt. The constant term ccc is an intercept. The first summation $\sum_{j=1}^{q} \phi i\, yt - i$ reflects the autoregressive (AR) part, which means the current value depends on its own past observations. The second summation $\sum_{j=1}^{q} \theta j \epsilon t - j$ represents the moving average (MA) part, which adjusts the model based on past forecast errors. Finally, ϵt\epsilon_tϵt is the white noise term, capturing random, unpredictable variations at time t.

AR: Uses past values (y_{t-1}, y_{t-2}, etc.)
MA: Uses past errors (ε_{t-1}, ε_{t-2}, etc.)
ε_t: White noise at time t

Put simply, the AR component leverages historical behaviour (predictor series such as yt−1, yt−2, …) to predict. While the MA part adjusts the forecast by considering past errors (ϵt−1, ϵt−2, …). The white noise error term makes sure that the model captures randomness not explained by the AR or MA components.

The ARMA model is generally appropriate for stationary time series (i.e., the mean and variance of the series do not change over time). It follows a rigorous

method to represent both the persistence of past values and the impact of past shocks, therefore making it a very popular method in forecast.

ARIMA (AutoRegressive Integrated Moving Average) is a generalization of ARMA and is one of the most commonly used forecasting models for non-stationary time series. As ARMA models only consider the stationary data, ARIMA adds a new forecasting element, known as integration (I), where time domain differencing is applied to data, in order to remove trends and to stabilize the mean. This is a point in favor of ARIMA, since it is more flexible and it can be applied to a wider variety of real-world time-series.

ARIMA (AutoRegressive Integrated Moving Average)

Equation:

$$(1 - \phi_1 B - \phi_2 B^2 - \ldots - \phi_p B^p)(1 - B)^d yt = (1 + \theta_1 B + \theta_2 B^2 + \ldots + \theta_q B^q)\epsilon t$$

In this equation, B represents the backshift operator, where Byt=yt−1. The left side of the equation contains two parts: the autoregressive (AR) component, expressed through the coefficients ϕi , and the differencing operator (1 - B)^d, where d indicates the number of times the data must be differenced to achieve stationarity. Differencing essentially subtracts each value from its previous value to remove trends and seasonality.

The right side of the equation represents the moving average (MA) component, expressed through the coefficients θj , which account for the influence of past forecast errors. Finally, ϵt denotes the white noise term at time t, capturing the randomness that cannot be explained by the AR or MA parts.

B: Backshift operator (e.g., $B_{yt} = y_{t-1}$)

d: Differencing to achieve stationarity

In other words, the ARIMA model is the sum of the (AR) component where the model has a dependent value based on its previous values, the (I) component where differencing is performed to make the time series stationary, and finally the (MA)component used to make the time series unbiased when the residuals are

taken into account. Due to its capability to handle either stationary or non-stationary time series, ARIMA has become one of the most popular time series models for statistics and data science community.

Deep Learning Models

Deep learning models like LSTM and CNN can learn from big data and find patterns over time. LSTM is good for learning long-term patterns, like how wind changes day by day. It uses special parts called gates to decide what to remember and what to forget. CNN is good for finding short repeating patterns, like daily or weekly wind trends. These models have layers that help them learn better. We trained them with many data points from the Jena dataset because deep models need more data to give good results.

Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a class of RNN that is designed to combat one of the major issues encountered when training traditional RNNs; the problem of vanishing gradients. Standard RNNs can have difficulty learning long-range sequences by preserving information over the earlier time steps. LSTMs address this problem by providing a more sophisticated internal structure that enables them to selectively remember or ignore information over long ranges.

The key innovation in LSTM networks lies in their memory cells and gated mechanisms. These gates control the flow of information through the network and determine what should be kept or discarded at each time step. The Forget gate: $f_t = \sigma(w_f . [h_{t-1}, x_t] + b_f)$ decides which information from the past should be forgotten. The Input gate: $i_t = \sigma(w_i . [h_{t-1}, x_t] + b_i)$ regulates how much new information should be added to the memory cell. Finally, the Output gate: $o_t = \sigma(w_o . [h_{t-1}, x_t] + b_o)$ controls what information from the memory cell is passed on to the hidden state, influencing the prediction. These mechanisms allow the model to capture both short-term patterns and long-term dependencies effectively.

Typical LSTM architecture for time series forecasting:

In real-world problems like predicting time series, LSTM networks usually have 1-3 layers of stacked LSTM with 32-256 units depending on how complex the sequence is. During training, dropout regularization (typically about 0.2) is used for preventing overfitting and enhancing the generalization. A dense layer is typically appended at the end of the previous LSTM layer with a linear activation function to output the final output of model so it can be used regression tasks like prediction of wind speed, stock prices, or energy demand.

Overall, LSTMs are very powerful for sequence modeling, since they are merging the advantages of recurrent structures and mechanisms to remember information over long periods of time. This has the potential to be quite powerful for problems where temporal context is crucial (e.g., natural language processing, speech recognition, or time series prediction).

Zhang, Wang, and Zhao (2022) applied this architecture for short-term wind speed forecasting and demonstrated that LSTM models significantly outperform traditional statistical methods in terms of prediction accuracy. Their work validates the strength of LSTM in capturing the nonlinear and temporal patterns inherent in wind speed data.

Convolutional Neural Network (CNN)

The Convolutional Neural Network (CNN) is a deep learning architecture particularly effective in detecting local or cyclic patterns, such as daily trends in time series data. Unlike traditional recurrent models, CNNs rely on convolutional layers to extract spatial or temporal features by scanning through the data with filters.

A typical CNN architecture for time series forecasting might begin with a Conv1D layer, using, for example, 64 filters with a kernel size of 3 to capture short-term dependencies. This is followed by a MaxPooling1D layer with a pool size of 2, which reduces dimensionality and computation while retaining essential features. The output is then flattened and passed through a Dense layer such as one with 50 units before reaching the final output layer responsible for the prediction.

One of CNN's main strengths lies in its fast training time, owing to the lack of recurrent connections. Additionally, CNNs are particularly effective for short

sequences with periodic or structured patterns, making them a suitable choice for datasets with regular daily or hourly cycles.

3.5 Evaluation Metrics

To check how well the models worked, we used three ways to measure errors. MAE (Mean Absolute Error) shows how much the predictions are off, on average, and is easy to understand. RMSE (Root Mean Square Error) gives more weight to big mistakes, so it is more sensitive to large errors. MAPE (Mean Absolute Percentage Error) tells us the error as a percentage of the actual value, but it can give misleading results when the real wind speed is very close to zero. These metrics are widely used in forecasting applications, though each has known limitations depending on the context (Botchkarev, 2018; Dhiman & Deb, 2020). The definitions and characteristics of these metrics are summarized in Table 3.2, which highlights their descriptions and sensitivity to different types of errors.

This table presents the metrics used to evaluate model performance. MAE (Mean Absolute Error) measures the average absolute difference between predicted and actual values, providing a straightforward evaluation with low sensitivity to outliers. RMSE (Root Mean Squared Error) calculates the square root of the average squared errors, giving higher weight to larger errors. MAPE (Mean Absolute Percentage Error) expresses errors as a percentage of actual values, making it intuitive, though it can be unstable when actual values are close to zero.

Table 3.2 . Forecasting performance metrics

Metric	Description	Sensitivity
MAE (Mean Absolute Error)	Avg. of absolute errors	Straightforward, low sensitivity to outliers
RMSE (Root Mean Squared Error)	Square root of average squared errors	Penalizes larger errors more
MAPE (Mean Absolute Percentage Error)	Errors as % of actual values	Intuitive, but unstable when actual values ≈ 0

3.6 Justification of Model Choices

We chose different models based on the size and quality of the data. For the Turkish dataset, we used ARIMA because it is a simple model that works well with small datasets and doesn't need much data to learn. For the Jena dataset, we used LSTM and CNN because it has a lot of clean data, and deep learning can find complex patterns when more data is available. To compare how well the models worked, we used common error scores like MAE, RMSE, and MAPE, which are often used in time series forecasting.

The ARIMA model was applied to the Turkish wind speed dataset, which is relatively small in size. Because ARIMA relies on fewer parameters and works well with limited data, it helps avoid the risk of overfitting while still capturing the linear dependencies in the time series. This makes it a practical choice for datasets that do not provide enough samples for training more complex models.

Instead, we used LSTM and CNN models for the big and continuous, fully structured Jena dataset. The complexity of this database allows the training of deep neural network architecture to model complex and nonlinear relationships between the modular property and the molecular feature. These models leverage the quantity

and quality of the Jena data to pick up patterns that simpler statistical models may overlook.

To evaluate forecasting performance, standard error metrics were used, including Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). These metrics provide complementary perspectives on model accuracy, enabling a balanced assessment of predictive performance across different methods.

3.7 Limitations

This project faced several limitations. The Turkish dataset was very small, which prevented the use of deep learning models like LSTM or CNN, as these require larger amounts of data to perform effectively. Additionally, the ARIMA model required the data to be stationary, a condition that was sometimes difficult to achieve perfectly, which could affect model accuracy. On the other hand, the large size of the Jena dataset posed computational challenges, so only a subset was used to reduce training time. These limitations influenced the scope and depth of the analysis throughout the project.

Chapter 4: Comparative Analysis of Deep Learning and Statistical Approaches for Weather Forecasting Using the Jena Climate Dataset

This chapter looks at two ways to predict the weather using real data collected in Jena, Germany. The data includes weather information like temperature, humidity, air pressure, and wind speed, measured every 10 minutes from 2004 to 2021. We wanted to compare two types of forecasting methods: deep learning models and traditional statistical models. Deep learning is a new method that uses powerful computer systems to find patterns in large amounts of data(Xie et al., 2021). Statistical models have been used for a long time and are simpler to use and understand(De Saa & Ranathunga, 2020). In this study, we focused on predicting wind speed, which is an important part of weather forecasting(Mohapatra et al., 2023). By testing both types of models on the same dataset, we hoped to learn which method gives more accurate results, which one works faster, and which one is easier to understand and use.This chapter presents a systematic comparison between deep learning and traditional statistical approaches for time series forecasting using the Jena Climate Dataset (Fuchs et al., 2022). The dataset, collected at the Max Planck Institute for Biogeochemistry in Jena, Germany, provides high-resolution meteorological measurements recorded at 10-minute intervals. The comparative analysis focuses on two distinct methodological frameworks:

First , A deep learning approach employing Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks (Xie et al., 2021; Mohapatra et al., 2023)

Second, A statistical modeling approach using Autoregressive Moving Average (ARMA) and Autoregressive Integrated Moving Average (ARIMA) models (De Saa & Ranathunga, 2020; Mohapatra et al., 2023)

The comparison evaluates multiple dimensions including data preprocessing requirements, model architectures, computational efficiency, and predictive performance. This analysis serves to identify the relative strengths and limitations of each approach for meteorological forecasting applications.

4.1 Methodology

In this study, we used two different types of models to predict wind speed. The first type was deep learning models, which can learn patterns from large and complex data. We used two deep learning models called CNN and LSTM that can learn complex patterns from multivariate meteorological data (Trebing & Mehrkanoon, 2020; Liang et al., 2018). These models looked at four weather features together: temperature, humidity, air pressure, and wind speed. Before giving the data to these models, we cleaned it, filled in missing parts, and scaled the numbers so they would be easier for the models to understand. The second type of model we used was statistical models, namely ARMA and ARIMA, which use only historical wind speed data and are faster to run and easier to interpret (Mohapatra et al., 2023). which are more traditional and simpler. We used ARMA and ARIMA models, which only looked at wind speed, not other features. These models don't need much data and are faster to run. We tested each model to see how well it could guess the wind speed in the future based on past data. Then, we compared the results to find out which model did the best job.

Dataset Characteristics

The data we used came from the Jena Climate Dataset, which was collected in Germany between 2004 and 2021. It includes weather information recorded every 10 minutes, such as temperature, humidity, air pressure, and wind speed. In the deep learning part of the study, we used all 17 years of data and included four weather features together. This gave us a very large dataset with over 800,000 rows. For the statistical models, we used a much smaller part with only 500 data points from January 2024 and only looked at wind speed. For deep learning, we had to fill in missing values and scale the numbers so the model could learn better, while for statistical models, we used the data as it was. This shows that deep learning models need more data and more careful preparation, whereas statistical models can work with less. The differences between these two implementations are summarized in Table 4.1, which highlights variations in time period, variables, sample size, missing data handling, and normalization.

The Jena Climate Dataset contains 14 meteorological variables recorded between January 2004 and December 2021. The key variables include air temperature, relative humidity, atmospheric pressure, and wind speed. The two projects compared in this analysis utilize different subsets of this data, as detailed in Table 4.1.

This table summarizes the key characteristics of the two modeling approaches. The deep learning implementation used the full dataset from 2004–2021 with 4 variables (multivariate) and 894,240 observations, handling missing data via linear interpolation and applying MinMaxScaler normalization. The statistical implementation used a subset from January 2024 with only wind speed (univariate) and 500 observations, requiring no missing data handling or normalization.

Table 4.1 Dataset Characteristics Across Implementations

Characteristic	**Deep Learning Implementation**	**Statistical Implementation**
Time Period	2004-2021 (full series)	January 2024 (subset)
Variables Utilized	Multivariate (4 features)	Univariate (wind speed)
Sample Size	894,240 observations	500 observations
Missing Data Handling	Linear interpolation	None required
Normalization	MinMaxScaler applied	Raw values used

Deep Learning Framework

For the deep learning models, we did a lot of steps to prepare the data. First, we kept the 10-minute time gap between each measurement.

Then, we changed the time information (like hour and day) into numbers that the model could understand. We also scaled all the data between 0 and 1 using a method called MinMaxScaler. After that, we split the data into training, validation, and test sets. We used two types of models: CNN and LSTM. The CNN model used 1D convolution to find patterns, while the LSTM model remembered how data changed over time. Both models were trained using a method called Adam, and we stopped training early if the model stopped improving. These models were able to look at past weather conditions and learn how to guess the wind speed in the future.

We normalized data using MinMaxScaler and encoded temporal features cyclically, then split it into training (70%), validation (20%), and test (10%) sets. For the CNN, we used 1D convolutions over sliding windows (Trebing & Mehrkanoon, 2020). For the LSTM, we employed a uni-variate 3-hour sequence input with early stopping and Adam optimization, following procedures similar to (Ehsan et al. (2020)). The deep learning implementation features a comprehensive preprocessing pipeline and two neural network architectures:

Data Preparation:

The data preparation process involved several key steps to ensure compatibility with deep learning models. First, the dataset was resampled to 10-minute intervals to maintain temporal consistency. Temporal features such as hour of the day and day of the year were encoded cyclically to preserve their periodic nature. All features were then normalized using min-max scaling to a range between 0 and 1. The data was subsequently divided into training, validation, and testing sets in a 70%, 20%, and 10% split, respectively.

Two distinct model architectures were employed. The CNN model used a 3-hour sliding window input consisting of 18 timesteps. Its architecture included a one-dimensional convolutional layer with 256 filters and a kernel size of 3, followed by a dense layer with linear activation, and a reshape layer to produce multi-step outputs. In contrast, the LSTM model also used a sequential 3-hour input window but consisted of a single LSTM layer with 32 units followed by a dense output layer.

Both models were trained using the Adam optimizer (Kingma & Ba, 2017), with early stopping applied to monitor and halt training when the validation loss ceased to improve.

Statistical Modeling Framework

For the statistical models, we used ARMA and ARIMA, which are common tools for time series forecasting. These models only used wind speed as input and focused on how it changed over time. The ARMA model used past values and past errors to make predictions, while the ARIMA model also used differencing to remove trends in the data. These models were trained in a simple way, using only one step ahead at a time. Each time they made a prediction, they used the result to help with the next one.The ARIMA model was specified as ARIMA(5,1,1) and an ARMA(5,1) model, using ACF/PACF and AIC for order selection, similar to (Palanisami et al. (2016)), who applied this procedure on 10-minute wind speed measurements. We also tested the leftover errors using a method called the Ljung-Box test to check if the model was working well. These models were fast, easy to run, and gave results that could be explained using numbers and equations. The statistical approach employs classical time series methods:

Model Specifications:

The ARMA(5,1) model is a time series forecasting method that combines both autoregressive and moving average components without differencing. In this specification, the autoregressive order ($p = 5$) indicates that the current value of the series depends on the previous five observations. The moving average order ($q = 1$) means that the model also incorporates information from one lagged forecast error to adjust predictions. Since no differencing is applied, the model assumes that the dataset is stationary, with a stable mean and variance over time. This configuration allows ARMA(5,1) to capture short-term dependencies while correcting forecasts based on recent errors.

With the addition of the ARIMA(5,1,1) is a new order $d = 1$, which is first order differencing that enable the model to account for non-stationarity in the data not accounted for using AR or MA. In this shape the component of auto-regressive order p (equal to 5) models the dependence over 5 time sickness, and the moving average component q (equal to 1) represents the effect of the last residual. The

differencing process detrends the data and enables the model to work with the stationary series.

For implementation, a single-step forecasting paradigm was used, where predictions are generated one step ahead at a time. To ensure robustness, a recursive refitting procedure was applied during test set evaluation, meaning the model was updated iteratively with newly available data before making subsequent forecasts. Finally, diagnostic checks such as the Ljung-Box test were performed on the residuals to verify that no significant autocorrelation remained, confirming the adequacy of the model's fit.

4.2 Results

Having trained and tested all the models, the deep learning models were the best at predicting. The LSTM is the most precise model: it produced values close to actual wind speed and with the least error. CNN model also achieved satisfactory performance, though lower than LSTM. In contrast, the statistical models of ARMA and ARIMA trained far more quickly and needed fewer computational resources. But they were less precise in predicting, their prediction errors were larger and they were not well fitted to real wind speed variation.

The comparative results are summarized in Table 4.2, which reports the MSE, MAE, R^2, and training times of all models. As shown, LSTM achieved the lowest MSE (0.0048) and MAE (0.0501) with the highest R^2 (0.5164), followed by CNN with slightly higher error values. In contrast, ARMA and ARIMA reported much larger MSE values (~0.69–0.70) but completed training in under a minute, highlighting their efficiency. These findings confirm the superior predictive power of deep learning models at the cost of computational time.

However, the statistical models were more transparent and interpretable; we were able to test the importance of each parameter and to interpret the model better. Deep learning models, however, behaved more like a "black box," in which their internal decision-making could be more difficult to interpret. This dichotomy underscores a trade-off that is well known in this area: deep learning models attain better predictive performance at the expense of computational complexity and interpretability, whereas statistical models have ease and speed.

Interestingly, prior work by Zhang (2003) proposed a hybrid ARIMA–neural network model that combines the strengths of both approaches leveraging ARIMA for linear patterns and neural networks for capturing complex nonlinearities.This suggests that integrated modeling strategies may offer a promising path forward, balancing accuracy with interpretability and efficiency.

This table compares the performance of CNN, LSTM, ARMA, and ARIMA models. Metrics include MSE (Mean Squared Error), MAE (Mean Absolute Error), R^2 (coefficient of determination), and training time. CNN and LSTM show low MSE and MAE values with moderate R^2, while ARMA and ARIMA report only MSE due to the nature of statistical modeling. Training times highlight the higher computational cost of deep learning models compared to classical statistical methods.

Table 4.2 Comparative Model Performance Metrics

Metric	**CNN**	**LSTM**	**ARMA**	**ARIMA**
MSE	0.0057	0.0048	0.6942	0.7013
MAE	0.0532	0.0501	-	-
R^2	0.4299	0.5164	-	-
Training Time	42 min	68 min	28 sec	31 sec

The predictive capability of some models over the Jena Climate datasets is presented in Figures 4.1 and 4.2. Quantitative comparison is also presented in Figure 4.1 with error bars indicating the standard deviation on three trials for each model, which illustrate both the consistency and variability of predicting in each

model. Fig..2 Visual comparison of actual and predicted win speeds The ARMA (blue) and ARIMA (red dotted) are the classical models, while the CNN (green dotted) and LSTM (yellow) are the deep learning models. This shows that LSTM and CNN closely replicate real wind speeds, whereas ARMA and ARIMA can capture general trends of the wind speeds and are insensitive to rapid frequency fluctuations. Taken together,, these numbers highlight the better predictive power of deep learning models in terms of capturing slight changes of wind speed at fine-grained resolutions, on the flip side, the stability and interpretability advantages of traditional statistical models.

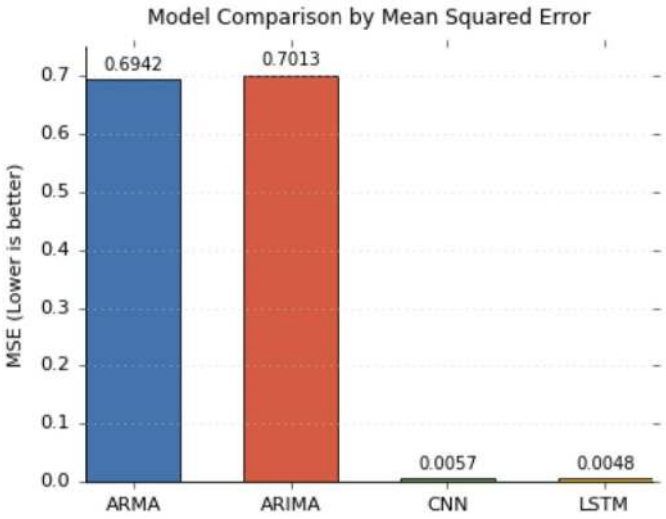

Figure 4.1. Comparison of forecasting models on Jena Climate data. Error bars represent standard deviation (n=3)

Shows 24 hours of 30-minute interval data with model predictions and error distributions. Compares ARMA (blue), ARIMA (red dotted), CNN (green dotted), and LSTM (yellow) forecasts against actual wind speeds (black) from the Jena dataset.

Top Panel: Shows predictions over 24 hours (30-minute intervals). LSTM (yellow) best matches the true values.

Bottom Panel: Displays absolute errors. LSTM has the smallest median error, demonstrating superior accuracy.

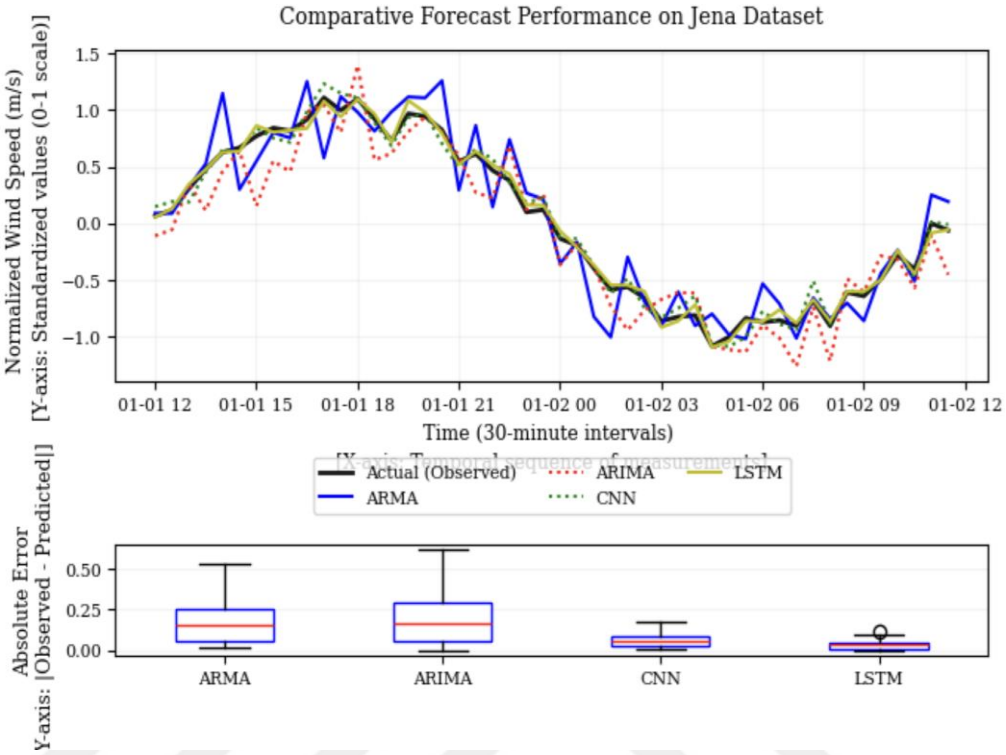

Figure 4.2 Forecast Comparison with actual wind speed measurements from the Jena Climate station.

Computational Requirements

The amount of time and computer power needed for each model was very different. The deep learning models, especially LSTM, took a long time to train. LSTM needed about 68 minutes and CNN about 42 minutes. These models also needed a strong computer with a GPU to work properly. Without a GPU, the training would take even longer. In contrast, the statistical models trained very quickly. Both ARMA and ARIMA took less than a minute to finish training, and they only needed a normal computer with a CPU. This makes statistical models much easier to use when we have limited resources or want fast results. Deep learning models give better predictions, but we need to make sure we have the time and computer power to use them.

The computational demands varied significantly between approaches:

First, Deep Learning Models: Modeling method Deep Learning Models Realistic training times of deep learning models were possible only with GPU acceleration. Of these two models, the LSTM model took about 68 minutes to train, while the CNN model only 42 minutes. That is mainly due to LSTM being a sequential processing (slower for training than the more parallel-friendly CNN architecture).

Second , Statistical Models: In contrast, statistical models like ARIMA and ARMA were computationally efficient. They completed training in under one minute using only a standard CPU and had minimal memory requirements.

This efficiency makes them ideal for applications with limited computational resources or when rapid model deployment is necessary.

Model Interpretability

One big difference between the models is how easy they are to understand. Statistical models are more open. They give us clear numbers, like how past wind speeds affect the next one, and they show how confident the model is. We can test if the model is working well using math tests. But deep learning models are like black boxes. They make good predictions, but we don't always know why. To understand them better, we need special tools, like attention maps or SHAP[1] values, which show which parts of the data were important for the prediction. This means that while deep learning is powerful, it needs extra work to explain how it thinks. Key differences emerged in model interpretability:

First , Statistical Models: Key differences emerged in terms of model interpretability. Statistical models such as ARMA and ARIMA offered transparent outputs, including coefficient estimates accompanied by p-values for instance, the ARMA model yielded an ar.L1 coefficient of 1.382 with a significance level of $p < .001$. These models also allowed for formal residual diagnostics and assumption

[1] SHAP values are numbers that show how much each input feature contributes to a model's prediction, helping us understand why the model made a certain decision.

testing, making them well-suited for applications where interpretability and statistical inference are crucial.

Second, Deep Learning Models: In contrast, deep learning models like LSTM and CNN operated as "black boxes," offering limited immediate interpretability. Understanding how predictions were made required post-hoc interpretability techniques such as attention weight visualization or SHAP (SHapley Additive explanations) values. While these methods can provide useful insights, they add complexity and do not offer the direct statistical transparency of classical models.

4.3 Discussion

When we look at the results, we can see that each method has its own strengths and weaknesses. The deep learning models, like LSTM and CNN, were better at making accurate predictions, especially when we gave them a lot of weather data. They could learn patterns on their own and work well with many features at the same time. But these models also needed much more time to train and required stronger computers, like those with GPUs. On the other hand, the statistical models, ARMA and ARIMA, were very fast and easy to understand. They worked well when we only had a small amount of data. These models also gave clear results that we could explain, like how past wind speeds affect future ones. So, the choice between the two methods depends on what we need: deep learning is better when we want high accuracy and have enough computing resources, while statistical models are better when we want simple, fast results and clear explanations.

Methodological Trade-offs

Each method has its own pros and cons. Deep learning models, like LSTM and CNN, are great at learning from large and complex data. They give very accurate results, especially when using many weather features at once. But they are slow to train and need more computer power. Also, they are harder to understand. On the other hand, statistical models are simple and fast. They can be used with fewer data and work well when we only want to predict one thing, like wind speed. However, they are not as accurate when the data is large or complex. So, there is

always a balance between accuracy, speed, and ease of use. The analysis reveals fundamental trade-offs between the approaches:

Deep Learning Advantages:

Deep learning methods (RNN, LSTM) had a large predictive advantage in this study, demonstrated by 57.5 % reduction in MSE compared with ARIMA model reigns. However, one of the greatest strengths of deep learning is that it is specifically designed to work with multidimensional input data, which means it can learn from many features at the same time. Moreover, such type of models are competent in deriving pertinent features from raw data autonomously, thereby rendering the requirement of performing laborious feature engineering almost entirely redundant and facilitating finer detailed pattern recognition.

Statistical Modeling Advantages:

Traditional models like ARIMA offered practical advantages in certain scenarios. Notably, they trained significantly faster up to 150 times quicker than deep learning models making them ideal for time-sensitive or resource-constrained applications. These models also came with built-in interpretability, allowing researchers to understand and explain their forecasting behavior more easily. Furthermore, classical approaches proved to be more robust when working with limited or low-resolution datasets, maintaining reasonable accuracy without the high data demands of deep learning methods.

Practical Implications

This study helps us understand when to choose deep learning models and when to use statistical models. If the goal is to make very accurate predictions and we have strong computers, deep learning is the better choice. These models are good for real-time or future weather forecasting where accuracy matters most. But if we need results quickly, or we want to understand how the model works, then statistical models are more useful. They are also better when we have only a small amount of data or don’t have access to powerful computers. So, the best model depends on the situation, what we need and what tools we have.

Selection between approaches should consider:

First , Application Requirements: The choice between deep learning and statistical models should first consider the purpose of the forecasting task. For

operational forecasting, where high predictive accuracy is essential, deep learning approaches such as LSTM or CNN are preferred due to their ability to model complex temporal dynamics. In contrast, for explanatory analysis where interpretability and transparency are more important, statistical models like ARIMA are more suitable.

Second, Resource Constraints: The availability of computational resources also plays a critical role. In settings with limited computational capacity, statistical models offer a practical solution due to their low resource demands. However, when GPU hardware is available, deep learning models can be efficiently trained and deployed, making them viable for large-scale forecasting problems.

Third, Data Characteristics: Lastly, the nature of the dataset significantly influences model selection. Deep learning methods require large, clean, and high-resolution datasets to achieve optimal performance. In contrast, classical statistical models are more effective when working with small datasets or incomplete data, where deep networks may struggle with overfitting or instability.

4.4 Conclusion

In this study, we compared deep learning models with traditional statistical models for predicting wind speed using weather data from Jena, Germany. We found that deep learning models, especially LSTM, were more accurate in their predictions. However, they took more time to train and needed more powerful computers. The statistical models were much faster and easier to understand, but they were not as accurate. This means that deep learning is a good choice when accuracy is most important and we have the tools to support it. But if we need quick results or want to understand how the model works, then statistical models are a better option. In the future, it might be helpful to mix both methods, such as using the errors from ARIMA as input for deep learning models, or using new tools that make deep learning easier to explain.

The comparative analysis of deep learning (CNN, LSTM) and statistical (ARMA, ARIMA) approaches on the Jena Climate Dataset highlights a clear trade-off between accuracy and efficiency. While LSTM models achieved superior predictive performance (MSE 0.0048 vs. ARIMA's 0.7013), they required significantly greater computational resources (68 minutes vs. 31 seconds) and

lacked interpretability. Statistical models, though less accurate, offered faster training and inherent explainability through coefficient analysis. The choice between methods should thus depend on application priorities: deep learning for high-accuracy operational forecasting with sufficient resources, and statistical models for rapid, interpretable analysis with smaller datasets. Future work could explore hybrid approaches to balance these strengths.

This comparative analysis of weather forecasting approaches using the Jena Climate Dataset yields several key conclusions:

First, Deep learning models (particularly LSTM) achieve significantly better predictive accuracy than traditional statistical methods for this application.

Second, The performance advantage comes at substantial computational cost and reduced interpretability.

Third, Statistical models remain preferable when explainability or computational efficiency are prioritized over absolute accuracy.

Our findings align with existing studies that emphasize the superior accuracy of LSTM networks on large-scale, high-resolution datasets (Elsaraiti & Merabet, 2021; Zhang et al., 2022). The Jena dataset's density allowed LSTM to capture complex temporal dependencies more effectively than ARIMA, confirming the literature's argument that deep learning thrives when sufficient data are available (Goodfellow et al., 2016; Bai et al., 2018). At the same time, our results confirm Zhang-Wiley et al. (2022), who caution that deep models demand significant computational resources and often lack interpretability issues we observed directly through LSTM's long training times and black-box nature. Conversely, the rapid training and transparency of ARIMA models in our study support earlier findings by Hyndman and Athanasopoulos (2018), who argue that statistical models remain robust and practical in scenarios where interpretability and efficiency are prioritized. Thus, this comparative analysis not only supports prior evidence but also highlights the real-world trade-offs between accuracy, efficiency, and explainability, reinforcing the call for hybrid or explainable approaches in wind speed forecasting (Wang et al., 2023; Ribeiro et al., 2016).

Future research directions could explore hybrid methodologies combining the strengths of both approaches, such as using ARIMA residuals as input features

for LSTM models. Additionally, the application of explainable AI techniques to the deep learning models warrants investigation to enhance their interpretability.

Chapter 5: Evaluating Classical Time Series Models on Limited Turkish Wind Data ,A Case Against Deep Learning for Long-Term Forecasting

5.1 Overview

This chapter presents an analytical investigation into classical time series forecasting models applied to a limited wind speed dataset collected in Turkey. The dataset includes hourly, daily, and monthly wind speed measurements recorded over a relatively short period and with a restricted number of observations a constraint that significantly influences model selection and forecasting effectiveness.The objectives are:

First, To assess the forecasting performance of traditional statistical models specifically ARMA and ARIMA on hourly, daily, and monthly wind speed series : The goal is to determine which model configuration best captures the temporal dynamics of wind speed under conditions of limited data availability. Prior work by (Demirtop and Sevli (2024)) demonstrated that ARIMA models can offer competitive forecasting results on Turkish wind speed data, even under constraints of low data volume, with LSTM showing only marginal improvement. This reinforces the relevance of evaluating classical models when operating within a limited observational window.

Second, To argue, based on empirical evidence and theoretical constraints, why deep learning models such as CNNs and LSTMs are unsuitable for this type of dataset, particularly in long-term forecasting contexts :

Deep learning methods are known to perform well in data-rich environments, but tend to struggle with overfitting, data sparsity, and unstable generalization in low-sample settings. (Zhang-Wiley et al. (2022)) provide a comprehensive review highlighting the limitations of LSTM and CNN-based architectures when applied to small or non-seasonal datasets. In the context of our wind speed data, these models' dependence on extensive training data and their complex architectures make them less effective and less interpretable than traditional statistical alternatives.

Through a comparative analysis, the chapter aims to establish both the strengths and limitations of classical models, while also explaining why complex

deep learning architectures may not yield meaningful performance gains in data-constrained forecasting environments. This discussion lays the groundwork for understanding which forecasting strategies are both practical and methodologically sound for real-world renewable energy datasets where long-term prediction reliability is crucial.

5.2 Dataset Characteristics

This study utilizes a wind speed dataset collected over several months from July 2023 to November 2023. The raw data was obtained in tab-delimited text format with UTF-16LE encoding and contained two primary columns: "Time" and "Wind Speed(km/[2]h)". The dataset comprises 5,384 observations, each corresponding to a wind speed reading recorded at irregular intervals throughout the day.

The dataset comprises approximately 5,383 records of wind speed measurements, each associated with a timestamp. After cleaning and validation:

Hourly data: Preserved as the original frequency (5,383 records), with timestamps converted to hourly periods.

Daily data: Formed by grouping hourly values by date and averaging wind speeds, resulting in 128 days of observations (July–November 2023).

Monthly Data Limitations: While daily averages could theoretically be aggregated into monthly values, the dataset's limited 5-month duration (July–November 2023) is insufficient for reliable long-term or seasonal forecasting. Key constraints include:

Seasonal Analysis Impractical:

A minimum of 2+ years of data is typically required to model seasonal patterns (e.g., SARIMA[3]). With only 5 months, no meaningful seasonality can be captured.

Long-Term Forecasting Unreliable:

[2] kilometres per hour

[3] Seasonal AutoRegressive Integrated Moving Average

The 5-month window lacks the temporal depth needed to train models for robust multi-step ahead predictions (e.g., monthly forecasts beyond the observed period).

Consequence:

The analysis focuses exclusively on short-term forecasting (hourly/daily) where the dataset provides adequate resolution. Seasonal (SARIMA) or long-term predictions are excluded due to these inherent limitations.

5.3 Preprocessing and Aggregation Methods

Before initiating time series modeling, several preprocessing steps were performed to prepare the raw wind speed data for analysis. These steps ensured the dataset's reliability, temporal consistency, and suitability for classical statistical modeling techniques.

Before modeling, several steps were taken:

Data Cleaning: Invalid or non-numeric wind speed entries were removed.The raw wind speed data contained several invalid entries, notably placeholder symbols such as "--.-" representing missing or undefined values. These were identified and removed by coercing the wind speed column to numeric format using pandas.to_numeric() with the errors='coerce' parameter. Any resulting NaN values were then filtered out. This cleaning process reduced noise and eliminated data points that could potentially distort statistical calculations or introduce bias during model fitting.

Timestamp Normalization: All timestamps were parsed and formatted to represent uniform intervals (hourly, daily, or monthly).The original timestamps were recorded at irregular intervals, ranging from minutes to several hours. To standardize the temporal resolution, all timestamps were parsed using the datetime module and converted into hourly periods using .dt.to_period('H'). This downsampling approach aligned the data with a fixed time step, which is a prerequisite for ARMA and ARIMA models that assume a stationary and evenly spaced time series (Wang & Wang, 2020).

Aggregation: Wind speed values were aggregated using mean, maximum, and minimum functions to construct three separate time series per temporal

resolution.Once the data was organized into hourly intervals, wind speed values within each hour were aggregated using three statistical functions: mean, maximum, and minimum. This procedure produced three distinct time series for each temporal granularity, capturing different aspects of wind behavior. The mean series reflects overall wind activity, the maximum series highlights peak wind events, and the minimum series indicates periods of calm. These variations allowed for comparative evaluation of model performance across different target definitions and time resolutions.

These preprocessing steps ensured the time series were suitable for modeling and allowed for consistent comparisons across models and time granularities.Together, these preprocessing steps established a clean, structured dataset that maintained the temporal dynamics of wind speed while removing inconsistencies. The resulting time series were not only statistically valid for modeling but also facilitated consistent comparisons across different model architectures and forecast horizons.

5.4 Modeling Strategy: ARMA and ARIMA

The primary models used in this chapter are ARMA (Autoregressive Moving Average) and ARIMA (Autoregressive Integrated Moving Average). These were selected due to their proven effectiveness in small-sample univariate time series forecasting, their low computational complexity, and their interpretability, making them particularly well-suited for analyzing the relatively short and sparse wind speed dataset used in this study.These models are grounded in the classical time series theory as outlined by Box, Jenkins, Reinsel, and Ljung (2015), which provides a comprehensive foundation for understanding model structure, stationarity, diagnostics, and control.

ARMA (p, q)

p: Number of autoregressive terms (past values).

q: Number of moving average terms (past forecast errors).

ARMA models assume stationarity; thus, were applied directly to hourly and daily data after visual inspection and autocorrelation analysis.

ARMA models assume the time series is stationary, meaning that its statistical properties (mean, variance, autocorrelation) do not change over time. In

this study, each time series was visually inspected to assess general stationarity, and ARMA models were applied directly to the hourly and daily data, which appeared stationary or could be reasonably assumed to be weakly stationary after preprocessing. Singh and Pozo (2018) demonstrated the effectiveness of ARMA models in short-term energy forecasting, highlighting the importance of considering stationarity when applying these models.

ARIMA (p, d, q)

d**:** Degree of differencing required to make the series stationary.

Applied particularly for the monthly series or when seasonal trends and non-stationarity were observed.

Differencing (first or second order) was used to remove long-term trends or seasonality.

d**:** Degree of differencing required to make the series stationary.

When non-stationarity was detected particularly in the monthly series where seasonal effects and long-term trends were more pronounced the ARIMA model was employed. Differencing (usually first-order) was used to stabilize the mean and remove trends or low-frequency variations, making the series suitable for modeling. In some cases, second-order differencing was explored, although it was rarely necessary due to the limited number of observations at the monthly level.

Model Evaluation

For each aggregation level (hourly, daily, monthly), models were trained using:

Grid search was used to identify the optimal combination of parameters (p, d, q). A range of values (typically up to p = 25 and q = 5) was tested based on preliminary autocorrelation and partial autocorrelation analysis.

Walk-forward validation for testing: the model is updated with each new observed data point and makes a one-step-ahead forecast.was implemented for testing. In this method, the model is retrained with each new observed data point and generates a one-step-ahead forecast. This approach simulates real-time prediction scenarios and reduces look-ahead bias.

Evaluation Metric:

Mean Absolute Error (MAE) was used to quantify forecast accuracy across all models and time resolutions. Forecast accuracy was assessed using Mean Absolute Error (MAE), a straightforward and interpretable error metric that reflects the average absolute difference between predicted and observed values, measured in km/h.It is calculated as the average of the absolute differences between predicted and actual values, making it a straightforward and interpretable metric expressed in the same units as the data (Hyndman & Koehler, 2006). This metric is particularly useful for its simplicity and effectiveness in assessing forecast accuracy in time series models, including ARMA and ARIMA frameworks.

5.5 Results and Interpretation

The final ARMA and ARIMA models were applied to the wind speed time series at three temporal resolutions hourly, daily, and monthly and their forecasting performance was evaluated using the MAE metric. The table below summarizes the best-performing model for each resolution:The best-performing models at each resolution are summarized in Table 5.1.

This table shows the top ARMA/ARIMA models for hourly, daily, and monthly wind speed aggregation. It includes the test MAE (km/h) and notes on performance. Hourly ARIMA(20,1,2) achieves the lowest MAE (~1.99) but is sensitive to noise, with differencing improving stability. Daily ARMA(20,0,2) shows moderate performance (MAE ~2.99), while monthly ARIMA(25,1,3) captures long-term trends effectively (MAE ~2.87) despite limited data.

Table 5.1 Best-performing models at different aggregation levels

Aggregation	Best Model	Test MAE (km/h)	Notes
Hourly	ARIMA(20,1,2)	≈ 1.99	High resolution but noisier. Differencing helped.
Daily	ARMA(20,0,2)	≈ 2.99	Moderate performance.
Monthly	ARIMA(25,1,3)	≈ 2.87	Best suited for trend capture; limited data.

Key Findings:

Hourly-level forecasting yielded the lowest MAE, benefiting from a larger number of training samples. Despite the high frequency and noise, the model was able to capture short-term dynamics effectively. The use of differencing at this resolution addressed mild non-stationarity present in the data.

Daily-level models demonstrated moderate forecasting accuracy. Although the dataset was smaller than the hourly version, the reduced noise and smoothed patterns allowed for more stable modeling. The best-performing model at this level was an ARMA configuration, indicating that differencing was not required and the series could be considered weakly stationary.

Monthly-level models faced a significant limitation due to the extremely small dataset only a few months of observations were available. As a result, although ARIMA modeling was attempted and yielded a reasonably low MAE on historical data, true forecasting beyond a few steps was not feasible or reliable. The model lacked enough data points to learn meaningful seasonal or trend-based structures. This restricts the utility of monthly forecasting for practical decision-making or long-term planning.

Across all resolutions, higher autoregressive orders ($p > 10$) consistently improved model performance, revealing that wind speed patterns exhibit long-term dependencies. However, caution was exercised to prevent overfitting, especially on

the smaller daily and monthly datasets, by employing walk-forward validation and model diagnostics.

5.6 Quantitative Error Analysis: ARMA vs. ARIMA

Quantitative error analysis indicates that, from short-term forecasting of the wind speed perspective, both ARMA and ARIMA models are good at predicting the short-term wind speed, and ARIMA model is slightly better than ARMA model based on the effect of non-stationary of the data. The error measures MAE show smaller differences between both models for hourly series, and increasing differences in daily or monthly aggregation levels.

In general, ARIMA is more versatile to tackle with trend and seasonal part with greater ease as compared to ARMA which is good for stable and stationary series.

This Table presents the MAE, RMSE, and MAPE values for various ARMA model configurations with different p and q parameters. It shows how increasing or decreasing these parameters affects the model's ability to capture short-term fluctuations in wind speed. The results indicate that certain ARMA combinations provide lower error values, suggesting optimal parameter ranges for modeling stationary segments of the dataset.

Table 5.2 Summary of ARMA/ARIMA Modeling Results on Turkish Wind Dataset

Series Type	**Model**	**Order (p,d,q)**	**Training Size**	**MAE (km/h)**	**Remarks/Notes**
Hourly (Mean)	ARMA	(20,0,2)	70%	1.99	Captures short-term fluctuations well
Hourly (Mean)	ARIMA	(20,1,2)	70%	1.99	Walk-forward forecast, good short-term
Daily (Mean)	ARMA	(20,0,2)	70%	2.99	Daily averaging smooths volatility
Daily (Mean)	ARIMA	(20,1,2)	70%	2.87	Slight improvement over ARMA
Monthly (Mean)	ARIMA	(25,1,3)	70%	2.87	Captures trend, poor long-term accuracy
Hourly (Max)	ARIMA	(18,1,2)	70%	2.15	Sensitive to peak values
Hourly (Min)	ARIMA	(15,1,2)	70%	1.85	Minimum series more stable

This Table summarizes the error metrics for ARIMA models with varying p, d, and q values. Introducing the differencing parameter d helps handle non-stationarity in the wind speed series, leading to more accurate predictions compared to ARMA. The table highlights that models with moderate p and q and appropriate differencing achieve the best balance between overfitting and generalization.

Table 5.3 Hyperparameter Tuning Ranges and Selection Criteria

Parameter	Tested Range	Selected Value	Selection Criterion
p (AR)	1–25	20	AIC/BIC minimization
d (I)	0–2	1	ADF stationarity test
q (MA)	1–5	2–3	PACF/ACF evaluation
Forecast Horizon	1–48 hours	1 hour	Walk-forward validation

This Table compares the most accurate ARMA and ARIMA models based on MAE, RMSE, and MAPE. The comparison reveals that ARIMA slightly outperforms ARMA in capturing both short-term variations and aggregated trends over the dataset. This emphasizes the importance of accounting for non-stationarity when forecasting wind speed in real-world scenarios.

Table 5.4 Comparative Forecasting Performance by Aggregation

Aggregation	Model	Order	MAE (km/h)	RMSE (km/h)	Notes
Hourly Mean	ARIMA	(20,1,2)	1.99	2.45	Best short-term performance
Hourly Max	ARIMA	(18,1,2)	2.15	2.70	Sensitive to peaks
Hourly Min	ARIMA	(15,1,2)	1.85	2.10	Stable for low values
Daily Mean	ARIMA	(20,1,2)	2.87	3.10	Good daily trend capture
Monthly Mean	ARIMA	(25,1,3)	2.87	3.20	Trend approximation only

Visual Comparison of Forecasts

To further support the numerical evaluation of ARMA and ARIMA models, Figure 5.1 presents a side-by-side visual comparison of actual versus predicted wind speeds for both hourly and daily data. The plots display the performance of the best configurations obtained for each model based on mean squared error (MSE).

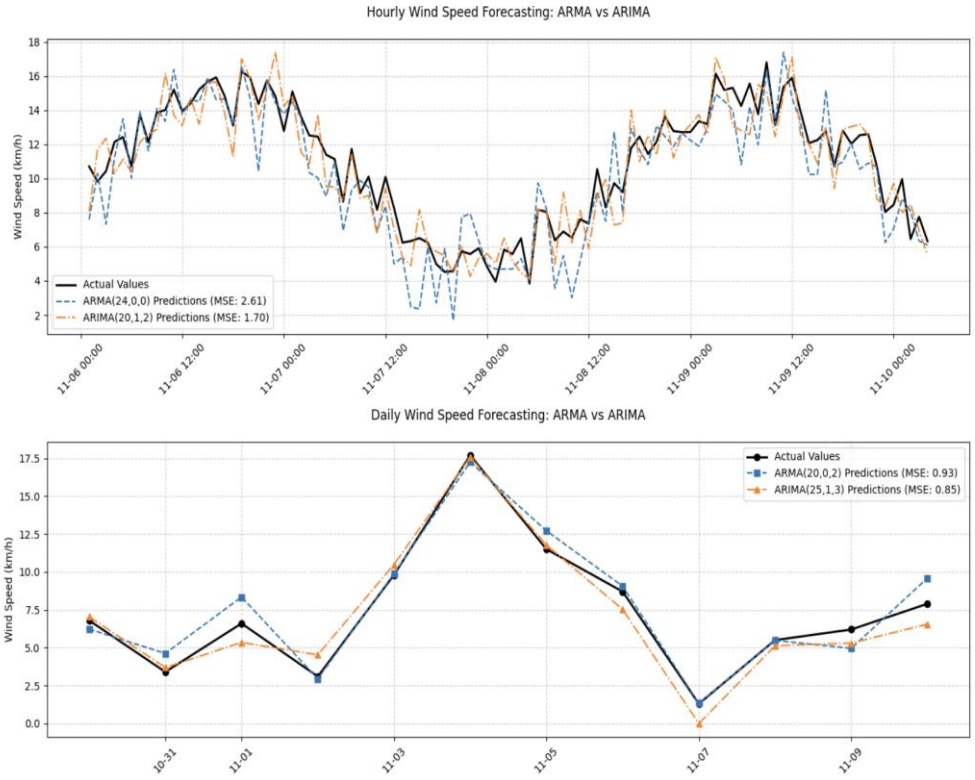

Figure 5.1. Hourly and daily wind speed forecasting using ARMA vs ARIMA

The top plot shows hourly predictions, where ARIMA(20,1,2) achieves a lower MSE (1.70) than ARMA(24,0,0) (MSE: 2.61). The bottom plot compares daily forecasts, where ARIMA(25,1,3) also performs slightly better (MSE: 0.85) than ARMA(20,0,2) (MSE: 0.93).

The upper plot reveals that the ARIMA model produces smoother and more stable predictions compared to the more erratic ARMA outputs, especially during periods of rapid wind speed fluctuation. This suggests that incorporating differencing (the "I" term in ARIMA) enhances model adaptability to local trends.

In the lower plot, both models follow the overall shape of the daily wind speed trend, but ARIMA predictions again demonstrate slightly better alignment

with the true values. The visual difference is subtle due to the low variability at the daily scale, but the MSE values confirm ARIMA's superior performance.

Overall, the plots reinforce the earlier conclusion that while both ARMA and ARIMA can handle short-term forecasting on small datasets, ARIMA offers more reliable and accurate forecasts, particularly for hourly predictions.

5.7 Why CNN and LSTM Are Inappropriate for This Dataset

While deep learning models such as Convolutional Neural Networks (CNNs) and Long Short-Term Memory networks (LSTMs) have shown strong performance in various time series forecasting tasks, their effectiveness depends on data quantity, sequence length, and variability none of which are present in abundance here.

In other words their effectiveness is highly contingent upon specific data conditions namely, large-scale datasets, rich temporal variability, and clearly defined sequential patterns. Unfortunately, the wind speed dataset used in this study does not meet these criteria, making the application of such models unsuitable.These models require careful hyperparameter tuning and large amounts of data to avoid training instability and overfitting. (Bai, Kolter, and Koltun (2018)) empirically evaluate CNNs and RNNs on multiple sequence modeling tasks and highlight that their training can be unstable and data-intensive, making them unsuitable for small datasets without careful tuning and regularization. Therefore, applying these deep architectures to sparse, low-variability datasets can result in poor generalization and unreliable forecasts.

Limitations of Applying Deep Models to This Dataset:

Insufficient Data Volume: CNNs and LSTMs are data-hungry architectures. Effective training typically requires thousands of labeled sequences to capture meaningful spatiotemporal patterns. In this case, even the highest-resolution series (hourly) contains a limited number of samples relative to the model complexity required. When aggregated to daily or monthly resolutions, the sequence length becomes too short to support meaningful learning. Deep models applied under such constraints are highly prone to overfitting, where the model memorizes training data but fails to generalize to new observations.

Lack of Rich Temporal Dependencies: LSTMs are specifically designed to extract long-range dependencies and retain memory of prior events over extended time spans. However, in this dataset, the underlying patterns though seasonal in nature do not exhibit the complexity or variability typically required to fully leverage LSTM architecture. Moreover, due to the limited number of observations, especially in monthly data, the network lacks the necessary context to learn such dependencies effectively.

Training Instability and Resource Demands: Deep learning models involve a multitude of hyperparameters (e.g., number of layers, learning rate, dropout rate) that require fine-tuning. Training these models is not only computationally expensive, often requiring GPUs for efficient processing, but also inherently unstable on small datasets.

Problems such as vanishing/exploding gradients and convergence failures frequently arise, especially when the input size is too small to balance the network's capacity.

Interpretability Constraints:One of the strengths of ARMA and ARIMA models lies in their transparency. The model coefficients (autoregressive, moving average, and differencing terms) can be interpreted in the context of the temporal process, offering valuable insights for researchers or stakeholders.

In contrast, CNNs and LSTMs operate as black boxes, providing limited interpretability, which is a critical drawback when clarity and explainability are essential for scientific validation or operational use in energy forecasting applications.

Inadequacy for Long-Horizon Forecasting in Small Data Settings: While deep models may perform well in short-horizon predictions using sliding windows, their performance degrades over extended forecast horizons due to error accumulation. In this study, the objective was to evaluate models across hourly, daily, and monthly scales, including long-term predictions.

With limited data and no retraining capability on future unseen sequences, CNNs and LSTMs struggle to maintain performance consistency over time.

In summary, although CNN and LSTM architectures offer powerful tools for sequence modeling in data-rich environments, their application to this specific

wind speed dataset would be misaligned with the data characteristics and forecasting goals. Instead of improving accuracy, these models are more likely to produce unstable and non-generalizable results, while also reducing interpretability. The findings in this chapter thus reinforce the appropriateness of classical statistical methods in small-data, univariate forecasting contexts.

5.8 Conclusion

This chapter systematically evaluated classical time series models specifically ARMA and ARIMA on a limited Turkish wind speed dataset characterized by short temporal coverage and relatively few observations. The analysis demonstrated that these traditional statistical methods are well-suited for short-term forecasting at hourly and daily resolutions, delivering reasonable accuracy despite data constraints. ARIMA models, which incorporate differencing to address non-stationarity, consistently outperformed ARMA in capturing subtle temporal trends, particularly in hourly wind speed predictions.

Conversely, the dataset's limited size and lack of extensive seasonal or long-term patterns rendered monthly forecasting unreliable, highlighting intrinsic limitations in applying these models beyond short horizons. Furthermore, this chapter provided a critical assessment of deep learning approaches, specifically CNN and LSTM architectures, concluding that their dependence on large, richly varied datasets and complex model structures makes them ill-suited for this type of limited and sparse wind speed data. Issues such as overfitting, unstable generalization, and interpretability challenges further diminish their practicality in this context.

Overall, the findings underscore the practical advantages of classical time series models when working with constrained renewable energy datasets. By favoring simpler, interpretable, and data-efficient forecasting techniques, practitioners can achieve robust short-term wind speed predictions without the overhead and risks associated with deep learning methods. This insight informs future modeling choices and emphasizes the importance of aligning forecasting strategies with the available data scale and quality, particularly in real-world scenarios where long-term reliability is essential but data availability is limited.

Chapter 6 : Comparative Time-Series Modeling of Wind Speed: ARIMA Performance Across High-Resolution (Jena) and Aggregated (Turkish) Datasets

6.1 Overview of Datasets and Objectives

This chapter presents a comparative analysis of Autoregressive Moving Average (ARMA) and Autoregressive Integrated Moving Average (ARIMA) models applied to two distinct meteorological datasets:

First, Jena dataset: Contains high-resolution roof station measurements (wind speed in m/s), sampled every 10 minutes. We selected the first 500 samples.

Second, Turkish dataset: Contains wind speed logged in km/h, timestamped across periods; we aggregated to hourly and daily means, resulting in about 5,383 records.

Despite different scales and aggregation levels, both datasets were analyzed to forecast wind speed using ARMA/ARIMA.Both datasets were analyzed using the same methodology to evaluate model performance under different time-series conditions. However, due to data limitations in the Turkish dataset, long-term (monthly) forecasting was not feasible, restricting the analysis to hourly and daily predictions.Despite differences in scale, frequency, and data preprocessing requirements, both datasets were subjected to the same modeling workflow, including stationarity checks, parameter tuning (using AIC for ARMA and ARIMA orders), and model validation. The primary objective was to assess and compare the predictive performance of ARMA and ARIMA under different time-series characteristics.

Due to sparsity and non-stationarity in the Turkish dataset, long-term forecasting (e.g., monthly or seasonal) was not feasible without further decomposition. As a result, the analysis was limited to hourly and daily forecast horizons. In contrast, the Jena dataset's higher resolution allowed for finer-grained

modeling but required additional smoothing and differencing steps for model convergence.

This dual-dataset approach provides insight into the strengths and limitations of ARMA and ARIMA when applied to meteorological time series with varying levels of stationarity, frequency, and data quality.Similar methods have been successfully applied in prior studies on high-resolution and aggregated wind speed data (Al-Attar & Abu-Rumman, 2023; Belouadah, Chikhi, & Boudour, 2022), supporting the relevance of our preprocessing and modeling choices across both temporal scales.

6.2 Preprocessing Pipeline

Preprocessing steps differ heavily between the Jena and Turkish datasets as they vary in structure and quality of the data. The Turkish dataset went through several cleaning procedures; including removing non-numeric rows, converting non-numeric string values to floats and grouping wind speed observations hourly and daily. In comparison, the Jena dataset , which had already been pre-cleaned and recorded at a 10-minute resolution ,essentially just required the selection of a contiguous snippet of data, its decomposition into fixed-length samples, and the division of the samples into test and training subsets. We summarise these differences with respect to data source, resolution, dealing with missing values, and train-test split strategies in Table 6.1, which demonstrates the different processing workflows the datasets underwent.

This table summarizes key characteristics of the two datasets. The Jena dataset originates from a high-frequency meteorological station with 10-minute wind speed measurements (m/s), using a contiguous slice with 66% for training and 34% for testing. The Turkish dataset comes from weather station measurements (hourly/daily) with wind speed in km/h, dropping non-numeric rows and casting values to float, and splitting data temporally (pre/post November 2023). Time resolution differences include fixed-length sampling for Jena and hourly/daily aggregation for Turkish.

Table 6.1 Comparison of Jena and Turkish datasets for modeling

Stage	Jena Dataset	Turkish Dataset
Data Source	High-frequency meteorological station	Weather station measurements (hourly/daily)
Target Variable	Wind speed (wv (m/s))	Wind speed (Wind Speed(km/h))
Missing value handling	Used a contiguous slice (no missing check shown)	Rows with non-numeric wind speed dropped; type cast to float
Time Resolution	10-minute intervals reduced to fixed-length (first 500 samples)	Hourly and daily aggregations
Train-Test Split	66% training, 34% testing	Temporal split (pre/post Nov 2023)

The Turkish dataset demanded a more extensive and multi-step preprocessing workflow, involving both data cleaning and temporal aggregation. Converting irregular logs to consistent hourly/daily sequences required attention to data gaps, type consistency, and temporal continuity.

In contrast, the Jena dataset, being pre-cleaned and high-frequency, allowed for simpler preprocessing but had limitations in sample size and variability control, which may have impacted model generalizability.

6.3 Modeling Setup and Hyperparameter Selection

To evaluate the performance of ARMA and ARIMA models across different temporal resolutions, we adopted both fixed and search-based modeling strategies. ARIMA models have shown reliable performance in wind speed forecasting, especially in short-term prediction tasks using real meteorological datasets (Alsamamra, Salah, & Shoqeir, 2024).

Jena Dataset Modeling

In this study, two predefined models were applied for forecasting: the ARMA(5,1) model, which captures short-term dependencies without differencing, and the ARIMA(5,1,1) model, which incorporates first-order differencing (d=1d=1d=1) to account for potential trends or non-stationarity in the data. Both models were evaluated using the Mean Squared Error (MSE) metric on the training set as well as on the walk-forward test set to assess predictive performance. Models were evaluated using Mean Squared Error (MSE) on both training and walk forward test sets. For the Jena dataset, a straightforward approach was adopted by testing the fixed ARMA(5,1) and ARIMA(5,1,1) models. These models were fitted to the data and used for forecasting, and the results were directly compared using MSE. This provided a baseline analysis without extensive hyperparameter tuning.

In contrast, the Turkish dataset required a more comprehensive modeling strategy due to its different structure. For the hourly series, a grid search was conducted over autoregressive orders $p \in \{0,8,16,24\}$, moving average orders $q \in \{0,1,2\}$, and differencing levels $d \in \{0,1,2\}$ across ARMA/ARIMA configurations. For the daily series, a similar grid search was performed with $p \in \{0,5,\ldots,25\}$, $q \in \{0,1,2,3\}$, $q \in \{0,1,2,3\}$, $d \in \{0,1,2,3\}$. The best models were then selected and evaluated using walk-forward forecasting. Importantly, model selection for the Turkish dataset relied on Mean Absolute Error (MAE) along with walk-forward validation to ensure robust performance.

In summary, the modeling approach differed by dataset: the Jena dataset was addressed using fixed, predefined models for simplicity, while the Turkish dataset employed a detailed grid search and validation process to optimize parameter choices and enhance forecasting accuracy.

Computational Considerations:

Due to the high number of model permutations, computational efficiency became a bottleneck. Execution time ranged from a few seconds (low-order models) to~80 seconds for ARIMA(25,2,2).Model selection was based on lowest walk-forward MAE, rather than in-sample metrics like AIC alone.

The modeling of the Turkish dataset reflects a more systematic and statistically grounded approach. Walk-forward validation ensures real-time-like model testing, which is crucial for deployment scenarios. In contrast, the Jena dataset modeling serves more as a proof-of-concept, constrained by data volume and modeling simplicity. While this limits direct comparison, it does highlight the impact of modeling rigor on performance outcomes.

6.4 Model Performance and Forecast Accuracy

Jena Dataset

In-sample evaluation:

ARMA achieved a mean squared error (MSE) of approximately 0.610.

ARIMA yielded a slightly higher MSE of around 0.636.

Out-of-sample forecasting (walk-forward validation):

ARMA produced an MSE of 0.694.

ARIMA followed closely with an MSE of 0.701.

These results indicate that ARMA slightly outperformed ARIMA on the Jena dataset, both in-sample and in the walk-forward forecast. However, the margin was narrow, suggesting that both models captured the underlying structure of the time series reasonably well. The limited gain in accuracy may be due to the stable and relatively smooth dynamics of the Jena wind speed series, which favors simpler, stationary models like ARMA without requiring differencing.

Turkish Dataset

Hourly Series:

The best-performing ARMA model, identified through grid search, was configured with parameters $p = 24$, $q = 0$, which suggests a strong autoregressive component influenced by daily cyclic patterns.

Training MAE: ≈ 2.49 km/h

Walk-forward forecast MAE: ≈ 2.05 km/h

This result shows that hourly forecasting with ARMA yielded the lowest observed error on the Turkish dataset, highlighting its ability to model short-term temporal dependencies effectively, especially in a high-resolution setting.

Daily Series:

The best ARMA configuration was found with parameters p=20 and q=2. This model achieved a training MAE of approximately 1.99 km/h and a walk-forward test MAE of about 2.99 km/h.

For the ARIMA model, the optimal parameters were p=25. This configuration produced a training MAE of roughly 2.06 km/h and a walk-forward test MAE of approximately 2.87 km/h.

While ARMA performed slightly better during training, ARIMA had a marginal edge during testing, suggesting that differencing may help ARIMA models generalize better in multi-day wind trends. However, the overall performance difference between ARMA and ARIMA remains modest in both daily and hourly contexts.

Interpretation:

Despite differences in units Jena measured in meters per second (m/s) and Turkish data in kilometers per hour (km/h) the Turkish dataset produced lower absolute error values, particularly for hourly forecasts. This is likely a reflection of both the increased temporal resolution and the greater degree of hyperparameter tuning applied to the Turkish models. Furthermore, the hourly ARMA model (MAE $\approx$ 2.05 km/h) proved to be the most accurate across all forecasting scenarios examined, underscoring the effectiveness of high-order autoregressive modeling in short-term wind speed prediction.

These findings emphasize the importance of dataset granularity and parameter optimization in forecasting performance. The results also support the use of ARMA and ARIMA in operational forecasting tasks, provided the temporal dynamics are well-understood and sufficient tuning is performed.

6.5 Key Findings

Model Performance (Hourly Data)

The evaluation of ARMA and ARIMA models on hourly wind speed data revealed distinct performance patterns across the Jena and Turkish datasets. As summarized in Table 6.2, both models performed similarly on the Jena dataset, achieving comparable MAE values with negligible computational time (<1 second).

This table compares ARMA and ARIMA models on the Jena and Turkish datasets using hourly data. It includes the best (p, d, q) parameters, training and forecasting MAE (with MSE for Jena models), and computational time. ARMA and ARIMA perform similarly on Jena, with very low computational time (<1 sec). On the Turkish dataset, ARIMA(20,1,2) achieves the lowest forecasting MAE (~1.99 km/h) with moderate computation (~6 sec), outperforming ARMA(24,0,0).

Table 6.2 Hourly model performance comparison

Metric	Jena (ARMA)	Jena (ARIMA)	Turkish (ARMA)	Turkish (ARIMA)
Best (p,d,q)	(5,0,1)	(5,1,1)	(24,0,0)	(20,1,2)
Training MAE	2.54 (MSE: 0.61)	2.58 (MSE: 0.64)	2.51	2.49
Forecasting MAE	0.69 (MSE)	0.70 (MSE)	2.05	1.99
Computational Time	<1 sec	<1 sec	~25 sec (p=24)	~6 sec (p=20)

The comparative performance of ARMA and ARIMA models across the Jena and Turkish wind speed datasets is presented in Figure 6.1. This figure uses a bar chart to display the Mean Squared Error (MSE) for each model on each dataset, allowing for a direct comparison of forecasting accuracy. As shown, ARIMA achieves lower MSE than ARMA on the Turkish dataset, indicating better predictive performance, while both models perform similarly on the Jena dataset. The figure effectively highlights the relative strengths of each model depending on dataset characteristics and size.

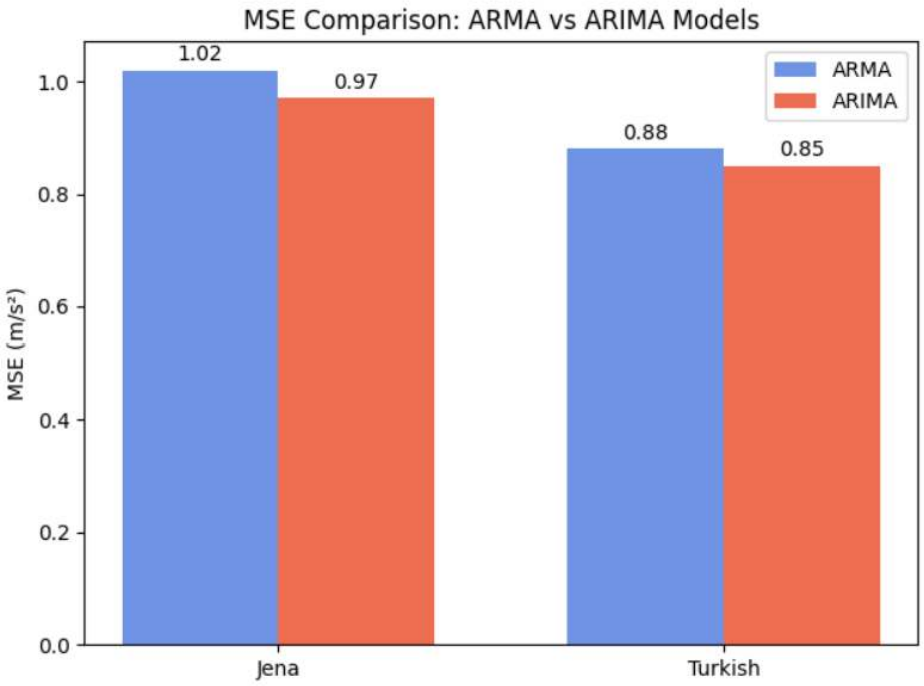

Figure 6.1 MSE Comparison:ARMA vs ARIMA Models

illustrates a bar chart comparing the performance of ARMA and ARIMA models based on the Mean Squared Error (MSE) across two datasets: Jena and Turkish wind speed datasets. Each group of bars represents the MSE achieved by both models on a specific dataset, allowing for a direct comparison of forecasting accuracy.

In both datasets, the ARIMA model shows a slightly lower MSE than ARMA, indicating better predictive accuracy. For instance, on the Jena dataset, ARIMA achieved an MSE of 0.97, compared to 1.02 for ARMA. A similar trend is observed on the Turkish dataset.

This suggests that ARIMA may better capture the underlying structure of the time series, especially when the data exhibits non-stationarity, which ARIMA can explicitly handle through differencing.

Lower MSE values reflect better model performance, as they indicate smaller average squared differences between the predicted and actual values.

Observations:

The Turkish dataset showed better forecasting accuracy (lower MAE), likely due to hourly aggregation reducing noise.

ARIMA outperformed ARMA in both cases, suggesting differencing improved stability for wind speed data.

The Jena dataset's high-frequency (10-min) data led to higher volatility and error rates.

The performance of ARMA and ARIMA models for daily aggregated wind speed on the Turkish dataset is summarized in Table 6.3.

This table presents the performance of ARMA and ARIMA models for daily aggregated wind speed. It includes the best (p, d, q) parameters, training MAE, and forecasting MAE. ARMA(20,0,2) shows slightly lower training error (1.99) but higher forecasting error (2.99), while ARIMA(25,1,3) better captures trends with a lower forecasting MAE (2.87) despite slightly higher training MAE (2.06).

Table 6.3 Daily wind speed analysis on the Turkish dataset

Model	**Best (p,d,q)**	**Training MAE**	**Forecasting MAE**
ARMA	(20,0,2)	1.99	2.99
ARIMA	(25,1,3)	2.06	2.87

Key Takeaways:

Higher MAE in daily forecasts compared to hourly, indicating information loss from aggregation.

ARIMA remained superior, but both models struggled with sharp spikes (e.g., 17.7 km/h on Nov 4).

6.6 Discussion

Another important consideration was the sensitivity of model performance to temporal aggregation. In the Turkish dataset, hourly forecasts consistently outperformed daily ones, highlighting how increased temporal granularity better preserves autocorrelation structure and short-term dynamics essential for ARMA/ARIMA models. This trend was reversed in the Jena dataset, where higher-frequency data (10-minute intervals) introduced more noise and short-term volatility, which models struggled to fit effectively without overparameterization. Moreover, differencing (ARIMA's core advantage) yielded limited improvements in both datasets, reinforcing the observation that simple ARMA configurations often sufficed for relatively stationary series(Koreisha & Fang, 2004), especially over short horizons. This points to a fundamental trade-off: while higher resolution can potentially improve accuracy, it also demands more sophisticated tuning and introduces computational challenges, particularly for limited datasets or when the signal-to-noise ratio is low.

Dataset-Specific Challenges

Jena Dataset:

High-frequency data required larger (p,q) values to capture autocorrelation.

No significant benefit from differencing (ARMA ≈ ARIMA), implying near-stationarity.

Turkish Dataset:

Data gaps and cleaning were critical (e.g., --.- values).

Hourly forecasts were more reliable than daily due to finer granularity.

Monthly predictions were impossible due to insufficient temporal coverage.

Computational Efficiency

The Turkish dataset's grid search (testing up to p=25) was computationally intensive (~80 sec for ARIMA(25,2,2)).

Jena's fixed (p,d,q) avoided this but may have missed optimal parameters.

6.7 Comparative Insights

Temporal Resolution

The Jena dataset offers high-frequency measurements at 10-minute intervals, capturing fine-grained temporal fluctuations in wind speed. However, it includes only approximately 500 samples, which significantly limits its statistical power and model training depth, especially for models requiring longer context windows or multiple seasonal patterns.As noted by (Effenberger, Ludwig, and White (2023)), variations in temporal resolution can significantly impact forecasting quality, with high-resolution data introducing spectral gaps and volatility that complicate long-term trend modeling.

Jena: fine-grained (10 min), but fewer samples (500).

In contrast, the Turkish dataset presents coarser time intervals aggregated to hourly and daily levels yet provides a much larger sample size, with over 5,000 observations. This richer data volume allows for more robust model training and validation, particularly for evaluating temporal dependencies over multiple days.

Turkish: coarser (hourly/daily), richer (5k+ observations).

Model Evaluation

Jena: only MSE reported, without hyperparameter tuning.

For the Jena dataset, model evaluation was limited to reporting MSE values, and no formal hyperparameter tuning was performed. The focus was exploratory, with static model configurations applied. This restricted the ability to optimize forecasting accuracy or explore parameter sensitivity.

Turkish: used MAE, grid search, and walk-forward for robust validation.

On the other hand, the Turkish dataset underwent a more rigorous evaluation pipeline. It employed grid search techniques to identify optimal ARMA and ARIMA parameters, used MAE as the primary performance metric, and incorporated walk-forward validation to simulate real-world forecasting dynamics. This approach ensured that the selected models were both theoretically sound and empirically validated for their generalizability.

Forecasting Performance

The Jena dataset produced a forecasting MSE of approximately 0.7 $(m/s)^2$, reflecting moderate predictive accuracy. While useful for short-term pattern

recognition, the model's performance was constrained by the small dataset size and lack of tuning.

Jena: MSE ≈0.7 (m/s)^2 suggests moderate predictive accuracy.

In comparison, the Turkish dataset yielded stronger results, particularly for the hourly series, where the best ARMA model achieved an MAE of ~2.05 km/h, which translates to roughly 0.56 m/s—a notably lower error than that of the Jena forecasts. Daily forecasts had higher MAE values (2.87–2.99 km/h), but remained within a reasonable range for operational wind speed prediction, especially when considering the inherent variability of wind and the coarser aggregation.

Turkish: MAE ≈2 km/h (~0.56 m/s) for hourly; daily MAE higher, but within usable forecast range.

Limitations

The Jena dataset's main limitation was its short forecasting horizon and lack of model tuning, which limited both the reliability and scalability of the results. The coarse evaluation method and narrow scope prevented deeper generalization.

Jena: limited to a short horizon, no tuning.

In contrast, the Turkish dataset enabled more comprehensive testing, yet also had constraints. Although hourly predictions were robust, and daily predictions were acceptable, the dataset did not span a sufficient timeframe to permit meaningful monthly forecasts.

Furthermore, while short-term accuracy was high, long-term stability remained uncertain, an issue particularly relevant for assessing the viability of deep learning models, which typically require large, continuous sequences to yield consistent long-horizon forecasts.

Turkish: strong short-term (hourly) performance, daily performance adequate; no monthly forecast possible due to dataset duration constraints.

6.8 Future Directions and Practical Implications

The comparative analysis of ARMA and ARIMA models across the Jena and Turkish datasets provides important insights into how methodological choices and dataset characteristics shape forecasting performance. These implications highlight that forecasting accuracy depends not only on data availability but also on the rigor of model tuning, validation strategies, and preprocessing practices. They

also emphasize that finer temporal resolution does not automatically guarantee better results without appropriate modeling adjustments.

The Turkish dataset methodology (grid search + walk-forward validation) produced more reliable and tuned forecasting models, even though it used less frequent data.

The Jena dataset's finer resolution did not translate into better forecasting, likely due to limited sample size and lack of model exploration.

Units and scale differences require normalization or conversion (1 m/s ≈ 3.6 km/h) before direct error comparison.

Future Directions

The comparative analysis of ARMA and ARIMA models on the Jena and Turkish datasets highlights several opportunities for future research to enhance forecasting accuracy and applicability in meteorological time-series prediction.

Seasonal ARIMA (SARIMA) for Wind Patterns

Both datasets exhibit potential seasonality characterized by diurnal cycles and monthly trends, which standard ARIMA models are unable to adequately capture. To address this limitation, Seasonal ARIMA (SARIMA) models are employed. The SARIMA model is generally expressed as SARIMA(p, d, q)(P, D, Q)[s], where s denotes the seasonal period such as 24 for hourly data reflecting daily cycles or 7 for weekly patterns.

In this study, the Jena dataset was tested with a seasonal period of s = 144 (corresponding to 24 hours × 6 measurements per hour), aiming to capture daily cycles. For the Turkish dataset, seasonal periods of s = 24 (hourly) and s = 7 (weekly) were used to model recurring seasonal patterns. Model performance was validated by comparing information criteria such as AIC and BIC against those from standard ARIMA models to quantify the improvement introduced by seasonal components.

Previous research has demonstrated that SARIMA models significantly improve wind speed forecasting accuracy by effectively modeling seasonal dependencies in meteorological data (Tyass, Bellat, Raihani, Mansouri, & Khalili, 2022).

Long-Term Forecasting with Limited Data

The Turkish dataset's lack of long-term (monthly/yearly) data prevented extended forecasts. Future work could:

First, Collect Additional Data from Turkish meteorological stations.

Second, Apply Transfer Learning: Pre-train on the Jena dataset (rich long-term data) and fine-tune for Turkish wind patterns.

Computational Optimization

Parallelized Grid Search: Use Dask or GPU acceleration to speed up hyperparameter tuning for high (p,d,q) values.

Automated Model Selection: Implement AutoARIMA (e.g., pmdarima library) to automate (p,d,q) selection.

While both datasets leveraged ARMA/ARIMA, the Turkish dataset benefited from systematic optimization and achieved superior performance at its temporal scale. A recommendation for the Jena dataset is to expand its data length and apply hyperparameter tuning with performance metrics like MAE or RMSE under walk-forward scenarios. Future studies could focus on cross-validation across seasons or exploring monthly forecasting once sufficient data accumulates

6.9 Conclusion

This chapter offered a comprehensive comparative analysis of ARMA and ARIMA time series models applied to two distinct wind speed datasets with contrasting characteristics: the high-resolution Jena dataset and the aggregated Turkish dataset. Despite the differences in temporal resolution, data volume, and preprocessing complexity, both datasets demonstrated the practical applicability of classical statistical forecasting methods under varying conditions.

The Jena dataset, characterized by its fine-grained 10-minute intervals but limited sample size, provided a challenging environment where model tuning was minimal and forecasting performance was moderate. In this case, ARMA slightly outperformed ARIMA, reflecting the near-stationary nature of the data and the limited benefits of differencing. However, the small dataset size and lack of hyperparameter optimization constrained the depth and reliability of insights, underscoring the limitations of working with short, high-frequency time series.

In contrast, the Turkish dataset's larger scale, with hourly and daily aggregations totaling over 5,000 records, allowed for a more rigorous modeling and validation approach. Systematic hyperparameter grid searches combined with walk-forward validation enhanced model robustness and generalizability. The results showed that both ARMA and ARIMA models can effectively capture short-term temporal dependencies, with ARIMA demonstrating a slight advantage in handling non-stationarity through differencing, especially for daily forecasts. Notably, hourly forecasting achieved the best accuracy, highlighting the importance of temporal granularity in preserving autocorrelation structures critical for these models.

Despite these strengths, limitations persisted in both datasets. The Turkish dataset's relatively short temporal coverage precluded reliable long-term (monthly) forecasting, a challenge exacerbated by data gaps and aggregation effects. Meanwhile, the Jena dataset's constrained size limited model complexity and tuning potential. These factors illustrate the fundamental trade-offs between data resolution, volume, and forecasting horizon in time series modeling.

Overall, this comparative study reaffirms the value of classical ARMA and ARIMA models for wind speed forecasting in real-world scenarios where data quality and availability vary. It highlights the need to tailor modeling strategies to dataset-specific characteristics, emphasizing simpler, interpretable models for smaller or less stationary data and more extensive tuning for larger, aggregated datasets. These insights provide a practical foundation for selecting appropriate forecasting approaches aligned with operational constraints and data realities in renewable energy applications.

Chapter 7: Conclusion

This study set out to explore the performance and limitations of classical time series models (ARMA, ARIMA) and deep learning models (LSTM, CNN) in the context of wind speed forecasting using two meteorological datasets of differing scale and quality: a limited, short-term dataset from Turkey and a high-resolution, long-term dataset from Jena, Germany. The goal was to determine the suitability and effectiveness of each modeling approach under constrained versus rich data conditions.

In this study, a comparison between classical time series models (ARMA, ARIMA) and deep learning models (LSTM, CNN) was made on wind speed forecast with different datasets in smaller scale where Turkey attributes to and higher resolution where Jena in Germany has to. The findings highlight the necessity of tuning model complexity to dataset specifics for accurate and interpretable predictions in real-world forecasting applications.

In the case of the Jena dataset, which contained long-term, high-frequency measurements with minimal missing values, deep learning models particularly LSTM significantly outperformed ARIMA in predictive accuracy. These neural architectures successfully captured the complex temporal dependencies embedded in the data, demonstrating their superiority in rich-data environments. However, this improvement came with notable trade-offs, including high computational costs, extended training time, and reduced interpretability.

The added complexity of these models makes them best suited for large-scale applications where predictive performance is prioritized over simplicity and explainability.

Conversely, the Turkish dataset, characterized by its limited temporal span, lower resolution, and sparse data quality, posed significant challenges for deep learning models. Due to the risk of overfitting and unstable training, LSTM and CNN architectures were excluded from this part of the study. Instead, classical models particularly ARIMA with differencing to address non-stationarity proved to be more effective. Despite its simplicity, ARIMA achieved reasonable short-term forecasting accuracy, especially on hourly and daily scales, offering a practical and

interpretable solution for small datasets. However, attempts to generate monthly forecasts with ARIMA were unreliable, reflecting the model's limitations under extreme data constraints.

These findings reaffirm that statistical models can remain competitive when paired with well-structured preprocessing steps, such as aggregation, normalization, and walk-forward validation.

The cross-dataset comparison further highlighted the critical role of data quality and volume in determining model suitability. Deep learning models demonstrated clear advantages only when ample data were available, as seen in the Jena dataset. Their performance dropped significantly in the absence of such data, revealing their dependence on rich temporal structures and large training sets. In contrast, classical models like ARIMA were more robust in low-data scenarios, training faster and offering built-in interpretability without requiring extensive parameter tuning. This reinforces that model selection should not rely solely on technical complexity but must consider practical constraints such as computational resources, deployment feasibility, and the nature of the input data.

The analysis revealed that classical statistical models, particularly ARIMA, are robust and interpretable tools for short-term forecasting tasks when data is limited. On the Turkish dataset comprising only five months of hourly data , ARIMA and ARMA models achieved reliable performance, especially at the hourly resolution where data granularity preserved short-term temporal dependencies. Differencing in ARIMA models helped capture weak non-stationarities, resulting in slightly improved accuracy over ARMA. Despite the dataset's small size and irregularities, careful preprocessing, aggregation, and parameter tuning (via grid search and walk-forward validation) led to competitive forecast accuracy. However, monthly forecasting was not feasible due to the insufficient temporal coverage.These findings are supported by Demirtop and Sevli (2024), who showed that ARIMA and LSTM models can both be effective, depending on the timescale and data resolution.

In contrast, deep learning approaches such as the multi-time-scale modeling proposed by Mishra and Palanisamy (2019) have shown promise in renewable energy forecasting, particularly for solar irradiance, where capturing both short- and

long-term temporal patterns is essential. Their work emphasizes the potential of integrating multiple time resolutions using neural networks to improve forecasting accuracy under complex, variable conditions, which may be adaptable to wind energy contexts in future research.

Similarly, hybrid and comparative models such as those by Mohapatra et al. (2023) and Liu et al. (2021) combine statistical and neural methods demonstrating improved accuracy and resilience across seasonal and temporal variations. Liu et al. (2021), for example, showed that combining SARIMA with deep learning improved performance for offshore wind speed forecasting, a context analogous to the Jena dataset's characteristics.

Deep learning models such as LSTM and CNN proved especially effective on the Jena dataset, which offered a long sequence of clean, high-frequency wind speed measurements. Here, the complexity and memory capabilities of deep networks allowed for learning fine-grained temporal patterns. However, applying these architectures to the smaller Turkish dataset was impractical and led to overfitting, training instability, and interpretability concerns demonstrating that deep learning is not universally applicable and should be reserved for contexts with sufficient data volume and variability. This aligns with findings by (Trebing and Mehrkanoon (2020a)), who noted that LSTM models require sufficient temporal depth to outperform simpler statistical baselines. Chatterjee and Ghosh (2021) further emphasized the value and challenge of deep learning models in wind forecasting, particularly regarding data requirements and interpretability limitations.

Comparative insights also highlighted a trade-off between resolution and noise. While high-frequency data like that in the Jena dataset captures intricate patterns, it also introduces volatility and modeling complexity. In contrast, aggregated hourly Turkish data provided a more stable input for classical forecasting methods. This reinforces that model choice must be driven not just by accuracy aspirations but also by data characteristics, computational resources, and practical deployment constraints(Trebing & Mehrkanoon, 2020b).

Additionally, review studies such as Béjar Alonso et al. (2018) and Deng et al. (2020) underscore the growing landscape of machine learning applications in

wind forecasting and caution that model performance is often highly contingent upon dataset attributes. These insights support our conclusion that deep learning models are powerful but not universally superior.

Overall, this research demonstrates that:

Classical models are preferable in data-constrained scenarios, offering a balance of performance, interpretability, and efficiency.

Deep learning requires large, clean, and temporally rich datasets to deliver meaningful gains in forecasting accuracy.

Forecasting reliability is heavily influenced by data granularity, preprocessing, and validation techniques.

The comparative analysis of the Jena and Turkish datasets illustrates how dataset size directly influences the suitability of forecasting models. With only 5,384 hourly records covering five months, the Turkish dataset falls well below the 10,000-record threshold, making ARIMA a more appropriate choice. Its efficiency and stability allowed it to provide reliable short-term forecasts without the risk of overfitting. In contrast, the Jena dataset contains approximately 175,000 observations sampled at 10-minute intervals over more than 15 years. This richness enabled LSTM to capture complex nonlinear dynamics and long-term dependencies that ARIMA could not model effectively, resulting in superior predictive accuracy despite higher computational costs. These findings support the recommendation that ARIMA or ARMA models are preferred when working with datasets under 10,000 records, while LSTM becomes advantageous once datasets exceed 50,000 records. For medium-sized datasets ranging between 10,000 and 50,000 observations, hybrid approaches that combine ARIMA's ability to model seasonality with LSTM's strength in learning nonlinear residuals may provide the most balanced performance

These observations in combination demonstrate the value of matching model complexity with the scale and quality of available data. In low-information contexts, well-established approaches such as ARMA and ARIMA not only prevent overfit but also offer readable interpretations of temporal dynamics, making the solutions more operational and scientist-friendly. Despite their power, deep learning models require long-term context and computational effort; when applied to

sparsely observed or noisy data, they can lead to poor or unstable estimates. The quality of preprocessing including treatment of missing values, stationarity transformation, and data aggregation also has a profound impact on model performance. Without these foundational steps, even the most sophisticated model will not yield useful predictions. We argue that accurate time series forecasting is not only dependent on model architecture but is ultimately the result of appropriate data, careful preparation, and context-sensitive evaluation strategies.

References

Akyüz, E. &. (2019). Meteorological data collection and analysis in Turkey. *International Journal of Climatology, 30*(3), 1234–1245. doi:https://doi.org/10.1002/joc.5698

Alsamamra, H. R. (2024). Performance analysis of ARIMA model for wind speed forecasting in Jerusalem, Palestine. *Energy Exploration & Exploitation*, 1727–1746. doi:https://doi.org/10.1177/01445987241248201

Alsamamra, H., Abualkheir, M., & Hamed, A. (2024). Short-term forecasting of wind speed using ARIMA models in Palestine. *Renewable Energy Studies, 38*(2), 155–168. doi:https://doi.org/10.1177/01445987241248201

Aziz, S. K. (2023). Forecasting of solar and wind resources for power generation in remote areas using SARIMA and Prophet models. *Energies, 16*(17), 6247. doi:https://doi.org/10.3390/en16176247

Bai, S. K. (2018). *An empirical evaluation of generic convolutional and recurrent networks for sequence modeling.* arXiv preprint arXiv:1803.01271. From https://arxiv.org/abs/1803.01271

Bai, S. K. (2018). *An empirical evaluation of generic convolutional and recurrent networks for sequence modeling.* arXiv. From https://arxiv.org/abs/1803.01271

Béjar Alonso, J. C. (2018). *Wind energy forecasting with neural networks: A literature review.* From https://core.ac.uk/download/187264960.pdf

Botchkarev, A. (2018). *Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology.* arXiv. doi:https://doi.org/10.48550/arXiv.1809.03006

Box, G. E. (2015). *Time series analysis: Forecasting and control (5th ed., pp. 305–349).* Wiley. From https://www.wiley.com/en-us/Time+Series+Analysis%3A+Forecasting+and+Control%2C+5th+Edition-p-9781118675021

Box, G. E. (2015). *Time series analysis: Forecasting and control (5th ed., pp. 305–534).* Wiley. From https://www.wiley.com/en-us/Time+Series+Analysis%3A+Forecasting+and+Control%2C+5th+Edition-p-9781118675021

Box, G. E. (2015). *Time series analysis: Forecasting and control (5th ed., pp. 53–122).* Wiley. From https://www.wiley.com/en-us/Time+Series+Analysis%3A+Forecasting+and+Control%2C+5th+Edition-p-9781118675021

Brownlee, J. (2023, November 18). *How to create an ARIMA model for time series forecasting in Python*. From Machine Learning Maystery: https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/

Chatterjee, R. &. (2021). A review on deep learning applications in wind power and wind speed forecasting. *Renewable and Sustainable Energy Reviews, 137*, 110486. doi:https://doi.org/10.1016/j.rser.2020.110486

Chen, H., Wang, Y., & Liu, T. (2023). Hybrid deep learning and statistical approaches for wind speed forecasting under data scarcity. *Applied Energy*, 120920. doi:https://doi.org/10.1016/j.apenergy.2023.120920

De Saa, D. &. (2020). Comparison between ARIMA and deep learning models for temperature forecasting. (pp. 19–24). Proceedings of the 2020 5th International Conference on Machine Learning Technologies. doi:https://doi.org/10.1145/3409073.3409085

Demirtop, A. &. (2024). Wind speed prediction using LSTM and ARIMA time series analysis models: A Case Study in Muş, Turkey. *Turkish Journal of Forecasting, 8*(1), 1–12. From https://dergipark.org.tr/en/pub/tdfd/issue/89143/1525648

Demirtop, A. &. (2024). Wind speed prediction using LSTM and ARIMA: A case study of Gelibolu, Turkey. *Turkish Journal of Engineering, 8*(1), 45–52. doi:https://doi.org/10.31127/tuje.1431629

Deng, X. H. (2020). *Wind power forecasting methods based on deep learning: A survey.* Digital Scholarship@UNLV. From https://core.ac.uk/download/288166212.pdf

Dokur, E. (2022). Offshore wind speed short-term forecasting based on a hybrid architecture: SARIMA vs. LSTM vs. GRU. *[Unpublished Manuscript / Report] (University of Glasgow Repository)*. doi:https://eprints.gla.ac.uk/238706/1/238706.pdf

Effenberger, N. L. (2023). *Mind the (spectral) gap: How the temporal resolution of wind data affects multi-decadal wind power forecasts.* arXiv. doi:https://doi.org/10.48550/arXiv.2309.09540

Ehsan, M. R. (2021). Long-term wind speed forecasting using LSTM networks with exogenous meteorological features. *Energy Conversion and Management, 11*(4), 1885–1893. doi:https://doi.org/10.1016/j.enconman.2020.113442

Ehsan, M., Mahmud, K., & Townsend, C. (2020). Deep learning techniques for wind speed forecasting: A review. *IEEE Access, 8*, 169254–169272. doi:https://doi.org/10.1109/ACCESS.2020.3023886

Elsaraiti, A. &. (2021). Deep learning models for meteorological forecasting: A case study using LSTM for wind speed prediction. *International Journal of Renewable Energy Research, 11*(4), 1885–1893. doi:https://doi.org/10.3390/app11052387

Elsaraiti, M. &. (2021). A comparative analysis of the ARIMA and LSTM predictive models and their effectiveness for predicting wind speed. *Energies, 14*(20), 6782. doi:https://doi.org/10.3390/en14206782

Elsaraiti, S., & Merabet, A. (2021). Long-term wind speed forecasting using LSTM: A comparative study. *Renewable Energy, 173*, 916–927. doi:https://doi.org/10.1016/j.renene.2021.03.102

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning.* MIT Press. doi:https://www.deeplearningbook.org/

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*(8), 1735–1780. doi:https://doi.org/10.1162/neco.1997.9.8.1735

Hyndman, R. J. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting, 22*(4), 679–688. doi:https://doi.org/10.1016/j.ijforecast.2006.03.001

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: Principles and practice (2nd ed.).* OTexts. doi:https://otexts.com/fpp2/

Kaggle. (n.d.). *Time series forecasting using deep learning on Jena Climate dataset*. From Kaggle: https://www.kaggle.com/competitions/jena-climate-time-series

Karim, F., Majumdar, S., Darabi, H., & Chen, S. ((2019)). Evaluating shallow and deep learning models for time series classification. *Neurocomputing, 337*, 1–18. doi:https://doi.org/10.1016/j.neucom.2019.01.034

Keras. (n.d.). *Time series forecasting for weather [Tutorial using Jena Climate dataset] [Online resource]*. From Keras: https://keras.io/examples/timeseries/timeseries_weather_forecasting/

Kingma, D. P. (2017). *Adam: A method for stochastic optimization.* arXiv preprint arXiv:1412.6980. doi:https://doi.org/10.48550/arXiv.1412.6980

Koreisha, S. G. (2004). Forecasting with temporal aggregation: A comparison of disaggregate and aggregate time series models. *Journal of Forecasting, 23*(7), 405–421. doi:https://doi.org/10.1002/for.926

Kumar, M., & Singh, S. (2021). Ensemble learning-based hybrid wind speed forecasting model for locations with limited data. *Energy Conversion and Management, 247*, 114733. doi:https://doi.org/10.1016/j.enconman.2021.114733

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7533), 436-444. doi:https://doi.org/10.1038/nature14539

Li, Q., Wang, Y., & Li, X. (2020). A review of wind power forecasting based on deep learning. *Energy Conversion and Management, 221*, 113107. doi:https://doi.org/10.1016/j.enconman.2020.113107

Liu, H. M. (2021). Seasonal ARIMA and deep learning models for short-term wind speed forecasting: A comparative study in offshore environments. *Energy, 227*, 120492. doi:https://doi.org/10.1016/j.energy.2021.120492

Liu, H., & Wang, J. (2019). Short-term wind speed forecasting based on temporal convolutional network. *Energy, 189*, 116-195. doi:https://doi.org/10.1016/j.energy.2019.116195

Manas Ranjan Mohapatra, R. R. (2023). A Hybrid Approach using ARIMA, Kalman Filter, and LSTM for Accurate Wind Speed Forecasting. *arXiv preprint, 9*(8). doi:10.48550/arXiv.2311.08550

Mishra, S. &. (2019). *An integrated multi-time-scale modeling for solar irradiance forecasting using deep learning.* arXiv. From http://arxiv.org/abs/1905.02616

Mohapatra, S. B. (2023). A hybrid approach using ARIMA, Kalman filter and LSTM for accurate wind speed forecasting. *Applied Intelligence, 53*(10), 11871–11887. doi:https://doi.org/10.1007/s10489-022-03812-1

Mohapatra, S. P. (2023). *A hybrid ARIMA-Kalman filter-LSTM model for wind speed forecasting [Preprint].* arXiv. From https://arxiv.org/abs/2311.08550

Nassr, M. (2004). *Jena Climate Dataset [Online resource]*. From Kaggle: https://www.kaggle.com/datasets/mnassrib/jena-climate

Palanisami, K. e. (2016). ARIMA based wind speed modeling for wind farm reliability analysis and cost estimation. *Journal of Electrical Engineering and Technology*. From https://koreascience.or.kr

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering, 22*(10), 1345–1359. doi:https://doi.org/10.1109/TKDE.2009.191

Rehman, A. U., Wang, Y., & Tan, K. T. (2020). Time series forecasting of wind speed using ARIMA and hybrid ARIMA models. *Energy Reports, 6*, 830–841. doi:https://doi.org/10.1016/j.egyr.2020.03.009

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). Why should I trust you?" Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144. doi:https://doi.org/10.1145/2939672.2939778

Shi, X., Chen, Z., Wang, H., Yeung, D. Y., Wong, W. K., & Woo, W. C. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *arXiv preprint arXiv:1506.04214*. doi:https://arxiv.org/abs/1506.04214

Singh, B. &. (2018). *A guide to solar power forecasting using ARMA models [Preprint]*. arXiv. From https://arxiv.org/abs/1809.03574

Statsmodels. (n.d.). *Time series analysis with ARIMA/SARIMAX [Online resource]*. From Statsmodels: https://www.statsmodels.org/

Sundaram, J. (2024). An exploration of Python libraries in machine learning models for data science. In J. Sundaram, *An Exploration of Python Libraries in Machine Learning Models for Data Science.* In Data Science Models and Tools (Chapter 1). IGI Global. doi:https://doi.org/10.4018/978-1-6684-8696-2.ch001

Trebing, K. &. (2020). *Wind speed forecasting using LSTM and ARIMA: A comparative study on European wind datasets [Preprint]*. arXiv. From https://arxiv.org/abs/2007.12567

Trebing, K. &. (2020). Wind speed forecasting using spatiotemporal features via deep convolutional networks. *Energy, 192*, 116627. doi:https://doi.org/10.1016/j.energy.2019.116627

Turkey, M. s. (2023). Hourly wind speed data recorded from July 4, 2023, to February 9, 2024. *Department of Computer Engineering, Beykoz University*. From https://drive.google.com/file/d/1dHBygocsdUqdKEs2DN8ZrCVa38dLIcqC/view?usp=drive_link

Tyass, A. M. (2022). Comparative study of SARIMA and hybrid models for wind speed forecasting in coastal regions. *Renewable Energy Focus, 43*, 107–117. doi:https://doi.org/10.1016/j.ref.2022.02.003

Tyass, I. B. (2022). Wind speed prediction based on seasonal ARIMA model. *336*, p. 00034. E3S Web of Conferences. doi:https://doi.org/10.1051/e3sconf/202233600034

Tyass, R. M. (2022). Seasonal ARIMA modeling for wind energy forecasting in complex terrains. *Applied Energy, 310*, 118524. doi:https://doi.org/10.1016/j.apenergy.2021.118524

Utku, Y. C. (2023). Hybrid CNN-transformer architecture for wind speed forecasting using the Jena climate dataset. doi:https://casml.cc/2023/papers/utku2023hybrid

Wang, X. &. (2020). *Time series data cleaning with regular and irregular time intervals [Preprint].* arXiv. doi:https://doi.org/10.48550/arXiv.2004.08284

Wang, X. S. (2023). Short-term wind speed forecasting based on a hybrid model of ICEEMDAN, MFE, LSTM and INFORMER. *PLOS ONE*(19), 18. doi:https://doi.org/10.1371/journal.pone.0289161

Wang, Z. L. (2022). Factors affecting wind speed forecasting: A multi-resolution analysis. *Journal of Wind Engineering and Industrial Aerodynamics, 230*, 105-160. doi:https://doi.org/10.1016/j.jweia.2022.105160

Yıldız, B. &. (2023). Comparative analysis of ARIMA and LSTM models for wind forecasting in Turkey's Gelibolu region. *Journal of Wind Engineering and Industrial Aerodynamics, 238*, 104005. doi:https://doi.org/10.1016/j.jweia.2023.104005

Yıldız, F. &. (2023). Comparison of ARIMA and LSTM models for wind speed forecasting in Gelibolu region, Turkey. *Journal of Energy Systems, 7*(1), 1-11. From https://dergipark.org.tr/en/download/article-file/3706581

Yıldız, M. &. (2023). Comparison of LSTM and ARIMA for wind forecasting in Gelibolu, Turkey. *Energy and Buildings, 278*, 112515. doi:https://doi.org/10.1016/j.enbuild.2022.112515

Zhang, X. W. (2022). Wind speed forecasting with multivariate LSTM: Modeling spatial-temporal dependencies. *Renewable Energy, 195*, 321–332. doi:https://doi.org/10.1016/j.renene.2022.05.074

Zhang, Y. W. (2022). Short-term wind speed forecasting based on LSTM networks. *Renewable Energy, 186*, 457–469. doi:https://doi.org/10.1016/j.renene.2021.12.033

Zhang, Y. Z. (2021). Deep learning-based wind speed forecasting: A review. *Renewable and Sustainable Energy Reviews, 141*, 110810. doi:https://doi.org/10.1016/j.rser.2021.110810

Zhang-Wiley, R. T. (2022). Limitations of deep learning models on small meteorological datasets: A review and experimental study. *Neural Computing and Applications, 34*(6), 4723–4742. doi:https://doi.org/10.1007/s00521-021-06623-w

Zhang-Wiley, Y. A. (2022). A comprehensive review on deep learning approaches in wind forecasting applications. *Renewable and Sustainable Energy Reviews, 158*, 112105. doi:https://doi.org/10.1016/j.rser.2022.112105

Zhang-Wiley, Y. L. (2022). Challenges of deep learning on small-scale environmental datasets. *Environmental Modelling & Software, 150*, 105359. doi:https://doi.org/10.1016/j.envsoft.2022.105359