Deep Learning-Based Forecasting of Electricity Demand and Carbon Emissions: Assessing the Impact of Renewable Energy Integration in the U.S. Power Sector

A Master's Thesis

Presented by
Ann Mary Thomas
Student Id :

Supervised by
Dr. Maitreyee Dey

In partial fulfillment of the requirement of
Master's in Data Analytics

Department of Computer Science and Applied Computing
London Metropolitan University, United Kingdom

Date: May 19, 2025

Acknowledgement

I would like to express my sincere gratitude to my supervisor, Dr. Maitreyee Dey, for her insightful guidance, continuous encouragement, and critical feedback throughout the course of this research. Her expertise and mentorship have been instrumental in shaping the direction and quality of this work.

I also extend my appreciation to the faculty members and academic mentors who have supported my intellectual growth, as well as to my friends and peers for their collaboration and thoughtful discussions that enriched this study.

Finally, I am deeply thankful to my family for their unwavering support, patience, and belief in my efforts during every stage of this academic journey.

Abstract

Accurate forecasting of electricity demand and associated CO_2 emissions is critical for effective energy management, grid reliability, and climate mitigation. Traditional models often address demand and emissions separately, overlooking their operational interdependence and the dynamic role of renewable energy integration. This study introduces a multi-output forecasting framework using Long Short-Term Memory (LSTM) networks to simultaneously predict electricity demand and CO_2 emissions. The model incorporates renewable energy penetration as a key input and leverages LSTM's ability to capture complex, nonlinear temporal dependencies without extensive manual feature engineering.

Comparative experiments with Recurrent Neural Networks (RNN) and a hybrid Restricted Boltzmann Machine plus Neural Network (RBM+NN) confirm the LSTM model's superior accuracy, especially in capturing seasonality and long-range demand-emissions dynamics. The framework is trained on a high-resolution, five-year dataset of hourly operational data from the U.S. power sector, focusing on California and Texas as case study regions.

Scenario-based simulations further demonstrate the environmental benefits of renewable integration. Under a 30% renewable penetration scenario, projected CO_2 emissions reductions reach up to 65.84% in California and 62.64% in Texas. These results validate the framework's applicability for decarbonization planning and highlight its policy relevance.

The research contributes a robust, scalable, and interpretable tool for integrated forecasting in energy systems. It equips planners and policymakers with actionable insights for aligning electricity generation strategies with sustainability goals.

Keywords: Electricity demand forecasting, CO_2 emissions prediction, Renewable energy, LSTM, AI-based modeling.

Contents

Acknowledgement **1**

Abstract **2**

1 Introduction **9**
1.1 Scope of the Study 11
1.2 Research Question 11
1.3 Objectives of the Study 11
1.4 Significance of the Study 12
1.5 Personal Development Reflection 12

2 Literature Review **14**
2.1 Traditional Approaches to Electricity Demand and Emissions Forecasting 14
2.2 Machine Learning and Hybrid Models in Energy Prediction 15
2.3 Deep Learning Architectures in Energy Forecasting 16
2.4 Renewable Energy Integration and Emissions Modelling 17
2.5 Identified Gaps and Research Motivation 18
2.6 Summary of Reviewed Literature 20

3 Methodology **21**
3.1 Data Understanding 23
3.1.1 Data Source and Description 23
3.1.2 Time Frame, Resolution, and Geographic Scope 23
3.1.3 Initial Data Exploration and Summary Statistics 24
3.1.4 Energy Mix Patterns and Emissions Contribution 25
3.1.5 Temporal and Regional Demand Patterns 26
3.1.6 Demand and Emissions Comparison Across Regions 29
3.1.7 Renewable Energy vs Emissions Trend Analysis 29
3.2 Data Preparation 31
3.2.1 Data Cleaning Approach 31
3.2.2 Handling Missing Values 31
3.2.3 Feature Normalization 32
3.2.4 Time-Based Feature Engineering 32
3.2.5 Lag Features and Rolling Statistics 33
3.2.6 Energy Mix Ratio Features 33
3.2.7 Final Data Transformation 34
3.3 Modeling 34
3.3.1 Model Selection Rationale 34
3.3.2 Long Short-Term Memory (LSTM) Networks 35

3.3.3 Recurrent Neural Networks (RNN) . . . 37
3.3.4 Restricted Boltzmann Machine (RBM) Combined with Neural Network . . . 39
3.3.5 Model Training Process . . . 42

4 Results and Analysis **45**
4.1 Evaluation Framework and Metrics . . . 45
4.1.1 Mathematical Formulation of Evaluation Metrics . . . 45
4.1.2 Rationale for Metric Selection . . . 46
4.2 LSTM Model Results and Analysis . . . 47
4.2.1 National-Level Forecasting Performance . . . 47
4.2.2 Regional Forecasting Performance: California and Texas . . . 48
4.2.3 Analysis of Regional Forecast Results . . . 50
4.2.4 Scenario-Based Simulation of Renewable Energy Impact on CO_2 Emissions . . . 51
4.3 RNN Model Results and Analysis . . . 55
4.3.1 Forecasting Performance of RNN Model . . . 55
4.3.2 Graphical Representation of RNN Forecasting Results . . . 55
4.3.3 Analysis of RNN Model Results . . . 56
4.3.4 Discussion on RNN Model Behavior . . . 57
4.4 RBM + Neural Network Results and Analysis . . . 58
4.4.1 Graphical Representation of RBM + NN Forecast Results . . . 58
4.4.2 Analysis of RBM + Neural Network Model Results . . . 59
4.4.3 Discussion on RBM + Neural Network Model Behavior . . . 59
4.5 Comparative Performance Analysis . . . 60
4.6 Interpretation of Model Performance . . . 61
4.7 Conclusion of Model Comparison . . . 61
4.8 Model Limitations and Generalizability . . . 62

5 Discussion **63**
5.1 Addressing the Research Question . . . 63
5.2 Interpretation of Model Behavior . . . 63
5.3 Significance of Renewable Integration Findings . . . 64
5.4 Practical Implications for Energy Policy and Planning . . . 65
5.5 Alignment with Existing Literature . . . 65
5.6 Study Limitations . . . 66

6 Conclusion and Future Scope **67**
6.1 Conclusion . . . 67
6.2 Research Contributions . . . 68
6.3 Ethical and Practical Considerations . . . 68
6.4 Future Work . . . 69

References **71**

Appendix **74**
Gantt Chart for Thesis Timeline . . . 74
Python Code for Data Cleaning . . . 74
Python Code for Feature Engineering . . . 75

Python Code for Visualization 76
Python Code for Modelling 81

List of Figures

1.1 Flow diagram showing the interaction between electricity demand, energy generation sources, and CO_2 emissions. 10

3.1 Adapted CRISP-DM Framework for Energy Demand and CO_2 Emissions Forecasting 22
3.2 Weekly Average Electricity Demand Over Time (2018–2023) 24
3.3 Electricity Demand, Net Generation, and CO_2 Emissions Over Time . . 25
3.4 Monthly Average Energy Generation by Source (2018–2023) 25
3.5 CO_2 Emissions Contribution by Source (National Level) 26
3.6 Electricity Demand Distribution by Day of the Week 26
3.7 Average Electricity Demand by Hour of the Day 27
3.8 Heatmap of Average Electricity Demand by Hour vs. Day of the Week 27
3.9 Heatmap of Average Electricity Demand by Hour vs. Month of the Year 28
3.10 California Hourly Demand Pattern by Month 28
3.11 Texas Hourly Demand Pattern by Month 29
3.12 Total Electricity Demand and CO_2 Emissions by Region 29
3.13 Historical Trend – Renewable Energy Share vs CO_2 Emissions (California) 30
3.14 Historical Trend – Renewable Energy Share vs CO_2 Emissions (Texas) 30
3.15 Internal Structure of an LSTM Cell Showing Forget Gate, Input Gate, Output Gate, and Cell State Flow 37
3.16 Unfolded Structure of a Simple RNN Across Three Time Steps 39
3.17 Conceptual Diagram of an RBM Combined with a Feedforward Neural Network 42

4.1 Forecasted vs. Actual Electricity Demand (National Level) 48
4.2 Forecasted vs. Actual CO_2 Emissions (National Level) 48
4.3 Actual vs. Predicted Electricity Demand - California 49
4.4 Actual vs. Predicted CO_2 emission - California 49
4.5 Actual vs. Predicted Electricity Demand - Texas 49
4.6 Actual vs. Predicted CO_2 emission - Texas 50
4.7 California – Historical Simulation of CO_2 Emissions with +30% Renewables 52
4.8 Texas – Historical Simulation of CO_2 Emissions with +30% Renewables 52
4.9 California – Future Forecast of CO_2 Emissions with +30% Renewables 53
4.10 Texas – Future Forecast of CO_2 Emissions with +30% Renewables . . . 53
4.11 Comparative CO_2 Emission Reduction (%) — Historical vs. Future, with 30% Renewables 54
4.12 Actual vs. Predicted Electricity Demand and CO_2 Emissions – California (RNN Model) 56

4.13 Actual vs. Predicted Electricity Demand and CO_2 Emissions – Texas (RNN Model) 56
4.14 Actual vs. Predicted Electricity Demand and CO_2 Emissions – California (RBM + NN Model) 58
4.15 Actual vs. Predicted Electricity Demand and CO_2 Emissions – Texas (RBM + NN Model) 59

6.1 Gantt Chart Showing Project Timeline and Phases 74

List of Tables

2.1 Summary of Key Literature on Electricity Demand and CO_2 Emissions Forecasting 20

3.1 Summary of Missing Data Handling Strategies 31
3.2 Summary of Temporal Feature Engineering for Forecasting 32
3.3 Summary of Lag Features and Rolling Averages Used in Forecasting . . 33
3.4 Energy Mix Ratio Features and Their Interpretations 33

4.1 Performance Metrics for Forecasting Models 47
4.2 Regional Forecasting Accuracy of LSTM Model (California and Texas) 50
4.3 CO_2 Emissions Reductions Under a 30% Renewable Scenario (LSTM-Based Simulation) 54
4.4 Forecasting Accuracy of RNN Model (California and Texas) 55
4.5 Forecasting Accuracy of RBM + Neural Network Model (California and Texas) 58
4.6 Comparison of Forecasting Models Based on Strengths and Weaknesses 61

Chapter 1

Introduction

The rapid increase in electricity consumption—fueled by urbanization, industrialization, and population growth—has introduced significant operational and environmental challenges for power systems. Accurate forecasting of electricity demand is essential to balance supply, prevent grid disruptions, and optimize resource allocation. Inaccurate forecasts can lead to energy waste, increased costs, and heightened risks of system instability. The interplay between growing electricity consumption, diverse energy generation sources, and the resulting carbon dioxide (CO_2) emissions underscores the critical need for intelligent forecasting systems that consider both environmental and operational constraints.

Figure 1.1 illustrates this interconnection, showing how different energy sources—fossil fuels and renewables—feed into electricity generation, which then contributes to electricity demand satisfaction and CO_2 emissions. This visual emphasizes the dynamic relationship between consumption behavior, generation portfolios, and environmental outcomes.

Electricity generation remains one of the largest contributors to global CO_2 emissions, primarily due to the dependence on fossil fuels such as coal and natural gas. As climate change intensifies, the importance of reducing CO_2 emissions has led to stricter environmental regulations, the adoption of carbon trading mechanisms, and the prioritization of sustainability planning. Accurate forecasting of both electricity demand and associated CO_2 emissions is therefore essential not only for maintaining system efficiency but also for supporting international climate commitments and decarbonization targets.

The global transition to renewable energy—particularly solar and hydro power—has emerged as a primary strategy to decouple electricity generation from CO_2 emissions. However, renewable energy sources introduce variability and uncertainty into the grid due to their dependence on environmental conditions. Forecasting in such contexts requires models that can handle complex temporal patterns and non-linear dependencies, particularly when integrating renewables into both demand and emission forecasting pipelines.

Conventional forecasting techniques such as Autoregressive Integrated Moving Average (ARIMA), Support Vector Machines (SVM), and Random Forests (RF) have

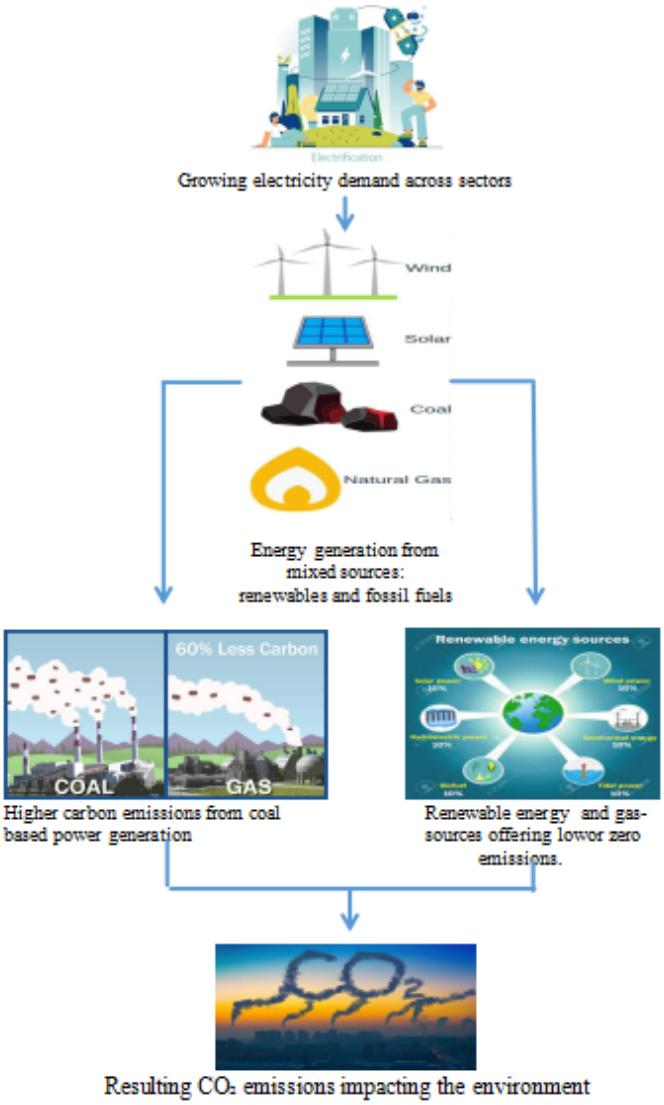

Figure 1.1: Flow diagram showing the interaction between electricity demand, energy generation sources, and CO_2 emissions.

shown effectiveness for short-term load predictions under relatively stable conditions. Yet, they often struggle with real-world energy systems due to their limited ability to capture long-range dependencies and multi-variable interactions. Moreover, these models traditionally handle electricity demand and CO_2 emissions as separate tasks, thereby missing the opportunity to model their interlinked behavior effectively.

Recent advancements in artificial intelligence (AI) and deep learning have introduced architectures capable of learning from multi-dimensional time series data. Among them, Long Short-Term Memory (LSTM) networks have proven especially powerful due to their internal gating mechanisms, which allow for learning long-term dependencies in sequential data. LSTM models have demonstrated success in energy forecasting tasks, but most studies treat electricity demand and emissions in isolation. The need for a unified model capable of forecasting both variables simultaneously has become more apparent in the context of decarbonization and grid modernization efforts.

This study proposes an integrated deep learning-based forecasting framework that uses an LSTM model to simultaneously predict electricity demand and CO_2 emissions. By incorporating features such as renewable energy penetration, lag-based demand behavior, and seasonality effects, the model provides a comprehensive tool for energy and emissions planning. To benchmark performance, the study also compares LSTM predictions with those generated by a Recurrent Neural Network (RNN) and a hybrid Restricted Boltzmann Machine combined with Neural Network (RBM+NN) model.

Unlike most prior studies that model electricity demand and CO_2 emissions as isolated forecasting tasks, this research proposes a novel multi-output deep learning framework that integrates both targets into a single predictive model. By combining temporal features with renewable energy penetration metrics, this unified architecture enables simultaneous forecasting while accounting for the dynamic interplay between energy consumption and environmental outcomes.

This study bridges the gap between electricity demand forecasting and emissions prediction by proposing a multi-output AI model that captures the co-dependent behavior of these variables. Unlike traditional approaches, the inclusion of renewable energy shares as dynamic inputs allows for scenario-based analysis of emission outcomes. The model's applicability across different regions further enhances its policy relevance, enabling grid operators, planners, and environmental agencies to make data-driven decisions. By demonstrating how renewable integration influences CO_2 emissions both retrospectively and prospectively, this research contributes valuable insights for sustainable energy transition and long-term environmental planning.

1.1 Scope of the Study

This research focuses on short- to mid-term forecasting of electricity demand and CO_2 emissions within the United States power sector, using high-resolution hourly data from July 2018 to June 2023. While national-level trends are acknowledged, the study centers on two regions—California (CAL) and Texas (TEX)—which represent contrasting generation portfolios and grid behaviors. The modeling framework is data-driven and relies exclusively on AI-based time-series methods, with an emphasis on LSTM networks configured for multi-output prediction. Additionally, renewable energy simulation scenarios are developed to evaluate emissions reduction potential.

1.2 Research Question

The central research question addressed in this work is: **How can AI-driven time-series forecasting improve the prediction of electricity demand and CO_2 emissions, enabling better energy management and sustainability planning?**

1.3 Objectives of the Study

The primary objective of this research is to develop an integrated, AI-driven forecasting framework capable of simultaneously predicting electricity demand and CO_2 emissions.

At the core of this framework is a multi-output Long Short-Term Memory (LSTM) network, designed to capture complex temporal dependencies inherent in energy system data. The model is configured to incorporate influential variables such as renewable energy penetration, lag-based demand history, and seasonality patterns, enabling it to learn the intertwined behaviors of electricity usage and emission outputs over time. In order to evaluate the effectiveness of the proposed approach, the LSTM model's forecasting performance is systematically compared against two baseline architectures: a conventional Recurrent Neural Network (RNN), and a hybrid model that combines a Restricted Boltzmann Machine (RBM) with a feedforward Neural Network (NN). These comparisons provide insight into the relative strengths and limitations of different time-series modeling techniques.

Beyond model development and comparison, the study also investigates the robustness of forecasting performance across regional contexts. California and Texas are selected as focal regions due to their contrasting energy generation profiles—California having a higher share of renewables, and Texas being more dependent on fossil fuels—allowing for a comparative assessment under diverse grid conditions. To further extend the practical relevance of the research, scenario-based simulations are conducted using the trained LSTM model. These simulations quantify the impact of increasing renewable energy contributions on CO_2 emissions by generating both retrospective (historical) and prospective (future) forecasts. Through these multi-faceted objectives, the study aims to provide a comprehensive, data-driven foundation for improving forecasting reliability and supporting emissions mitigation strategies within modern power systems.

1.4 Significance of the Study

This study bridges the gap between electricity demand forecasting and emissions prediction by proposing a multi-output AI model that captures the co-dependent behavior of these variables. Unlike traditional approaches, the inclusion of renewable energy shares as dynamic inputs allows for scenario-based analysis of emission outcomes. The model's applicability across different regions further enhances its policy relevance, enabling grid operators, planners, and environmental agencies to make data-driven decisions. By demonstrating how renewable integration influences CO_2 emissions both retrospectively and prospectively, this research contributes valuable insights for sustainable energy transition and long-term environmental planning.

1.5 Personal Development Reflection

Undertaking this thesis has been a pivotal experience in my academic and personal development. The interdisciplinary scope—spanning machine learning, energy systems, and sustainability forecasting—challenged me to acquire and apply advanced technical and analytical skills beyond traditional coursework.

Through independent management of this time-bound research, I developed strong project planning and organizational skills. Using the CRISP-DM framework and structured tools like Gantt charts enabled me to set clear milestones and maintain consistent

progress throughout the study.

My technical proficiency in Python programming, time-series modeling, and neural network design (LSTM, RNN) improved significantly. Moreover, I enhanced my capabilities in data cleaning, multi-output forecasting, and scenario analysis—skills directly applicable to real-world data science problems.

In parallel, the writing process deepened my academic communication abilities, including synthesizing literature, structuring arguments, and presenting findings clearly and professionally. Addressing ethical implications of AI in energy forecasting also expanded my understanding of responsible innovation and societal accountability.

This experience has equipped me not only with specialized knowledge but also with transferable competencies essential for professional growth. It has reinforced my motivation to contribute to data-driven environmental policy and continue exploring intelligent systems for sustainable development.

Chapter 2

Literature Review

2.1 Traditional Approaches to Electricity Demand and Emissions Forecasting

Electricity demand forecasting has long been an essential component of energy system planning, ensuring reliable supply, economic efficiency, and grid stability. Traditional statistical models such as the Autoregressive Integrated Moving Average (ARIMA) have been extensively utilised in this context. The ARIMA model, developed by Box and Jenkins, has served as a foundational tool for univariate time-series forecasting [1]. Its strength lies in modelling linear dependencies and capturing short-term temporal patterns. However, ARIMA and its variants often struggle with non-stationary data and are limited in their ability to accommodate multiple exogenous variables, making them less suitable for complex modern electricity grids [2].

Several studies have attempted to address these limitations through hybridisation. For instance, Zhang [3] proposed combining ARIMA with Artificial Neural Networks (ANNs) to model linear and nonlinear patterns simultaneously. This hybrid method improved prediction accuracy but introduced new challenges, particularly in model interpretability and hyperparameter tuning. In the context of CO_2 emissions, early approaches also relied heavily on econometric models or decomposition techniques such as Index Decomposition Analysis and the Divisia Index [4, 5]. These methods, while effective in understanding historical emissions drivers, lacked predictive capacity and were often static in nature, assuming fixed economic and structural relationships.

In terms of emissions forecasting, traditional regression-based models have been used to estimate carbon outputs based on variables such as fuel mix, energy intensity, and GDP growth. For example, Ang and Zhang [1] demonstrated the utility of decomposition analysis in identifying the contributions of different sectors and energy sources to emissions growth. However, such models are generally limited by their static assumptions and inability to capture time-varying interactions between demand, generation, and emissions. The lack of sequential learning capability restricts their effectiveness for real-time or scenario-based forecasting in dynamic energy systems [6].

Despite the foundational role of these classical methods, they are increasingly being outpaced by more advanced data-driven models that can handle multivariate, non-

linear, and sequential dependencies inherent in modern electricity grids. This growing demand for more flexible and adaptive forecasting tools has paved the way for machine learning and deep learning approaches, which have demonstrated considerable promise in recent years.

2.2 Machine Learning and Hybrid Models in Energy Prediction

As electricity systems have become more decentralised and influenced by variable generation sources such as solar and wind, the demand for models capable of capturing complex and nonlinear dynamics has increased. This has led to widespread interest in machine learning (ML) approaches, which can handle higher-dimensional datasets and learn intricate relationships among variables without requiring predefined functional forms.

Support Vector Machines (SVMs) and Support Vector Regression (SVR) are among the most widely applied ML methods for demand and emissions forecasting [7, 8]. SVR has demonstrated improved performance over classical models, especially in cases involving nonlinearity and high variance [9]. However, its performance is highly dependent on kernel selection and parameter tuning, and it lacks intrinsic support for sequential learning. Similar concerns apply to Decision Tree-based algorithms, such as Gradient Boosting Decision Trees (GBDT) and Random Forests, which have been used to model short-term electricity demand and associated carbon emissions [10, 11]. These models offer strong interpretability and robustness to outliers, but they typically require extensive feature engineering to account for temporal structure.

To overcome some of these issues, researchers have proposed hybrid ML models that combine statistical methods with neural networks or ensemble techniques. For example, the work of Wang et al. [12] and Amjady and Keynia [13] introduced ARIMA-SVR and ARIMA-ANN hybrids that significantly improved forecasting accuracy by leveraging the strengths of each component. Similarly, Khosravi et al. [14] proposed an integrated ML approach that fused SVMs with probabilistic forecasting techniques to quantify uncertainty. While these hybrid approaches yielded promising results, they still lacked the ability to directly model sequential data, necessitating lag-based inputs or manual temporal feature construction.

Random Forests, as studied by Chen et al. [15], offer an ensemble-based solution that aggregates multiple decision trees to mitigate overfitting. These models have been effective in short-term demand forecasting and emissions classification tasks. However, they struggle to extrapolate well to unseen data or novel scenarios unless carefully retrained and supplemented with exogenous features.

Another recurring limitation of conventional ML models is their static representation of features. Most do not evolve their feature understanding over time unless manually engineered. This limitation becomes pronounced in dynamic systems where factors like renewable penetration, policy regulations, and economic activity interact in non-linear and evolving ways.

In response, many researchers have turned to deep learning models that natively support sequence learning, thereby enabling the capture of time-dependent behaviours. These newer models mitigate the reliance on extensive preprocessing and enable more realistic, autonomous pattern recognition, setting the stage for the adoption of advanced architectures such as RNNs, LSTMs, and GRUs in energy forecasting.

2.3 Deep Learning Architectures in Energy Forecasting

With the increasing availability of granular, high-frequency energy data, deep learning has emerged as a powerful alternative to traditional machine learning approaches for forecasting applications. In contrast to classical and hybrid models that require extensive feature engineering, deep learning models are capable of learning temporal dependencies and complex feature representations directly from raw sequential data. Among these, Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, and Gated Recurrent Units (GRUs) have gained considerable attention in electricity demand and emissions forecasting research.

RNNs were among the earliest neural architectures applied to sequential forecasting tasks in the energy domain [16]. Their capacity to retain information through feedback loops allows them to model time-series data effectively. However, RNNs suffer from limitations related to the vanishing gradient problem, which inhibits their ability to learn long-range dependencies. This constraint makes them less suitable for tasks involving seasonal demand patterns or emissions linked to slow-changing generation trends.

To address this, Hochreiter and Schmidhuber introduced the LSTM network [17], a special form of RNN that integrates memory cells and gating mechanisms to control information flow across time steps. This architecture has been widely adopted in recent studies for short-term and long-term load forecasting [18, 5], CO_2 emissions prediction [19], and hybrid tasks involving both. LSTM models have shown superior performance in learning from multivariate time-series inputs, particularly when capturing non-linear and long-horizon dependencies.

Kong et al. [18] demonstrated the strength of LSTM in forecasting residential electricity consumption under varying weather and seasonal conditions, outperforming ARIMA and feedforward neural networks. Likewise, Cho et al. [20] proposed GRUs as a lighter alternative to LSTMs, with fewer trainable parameters and comparable performance. GRUs have been used effectively in cases with limited data or resource constraints, although they are still underexplored in the emissions forecasting context. Bidirectional LSTMs (Bi-LSTM) have also been employed to enhance model understanding by considering both past and future context within a sequence. Bedi et al. [21] applied Bi-LSTM to power load forecasting, demonstrating improvements in scenarios with strong periodic signals. Nonetheless, their computational overhead remains a challenge for large-scale, real-time applications.

In parallel, deep learning has also enabled novel hybrid architectures that combine temporal and spatial feature extraction. Singh and Suganthan [22], for instance, developed a CNN-LSTM hybrid to capture both short-term variations and spatial dependencies in industrial CO_2 emissions. Although their work was primarily focused on manufacturing processes, the methodology demonstrates the potential of deep neural networks for emissions-related tasks.

Multi-output forecasting using deep learning is another promising avenue, where models are trained to predict multiple interrelated targets—such as electricity demand and CO_2 emissions—simultaneously. Yin et al. [23] explored this through a multi-task LSTM architecture for electricity price and demand prediction. This aligns closely with the objectives of the current study, which seeks to extend this concept to include emissions forecasting alongside demand.

Despite these advancements, most existing applications focus on either demand or emissions in isolation. Very few studies have developed joint models that consider their interdependencies, particularly in relation to variable renewable generation. This presents a key research gap that the present study addresses by developing a multi-output LSTM forecasting framework, incorporating emissions and demand within a single temporal architecture while accounting for renewable share and temporal influences.

2.4 Renewable Energy Integration and Emissions Modelling

The integration of renewable energy sources into power systems is widely recognised as a cornerstone of global decarbonisation strategies. Solar and wind energy, in particular, offer emission-free electricity but present new forecasting challenges due to their intermittent and weather-dependent nature. Understanding how renewables influence both electricity demand and CO_2 emissions is crucial for the development of predictive tools that are aligned with sustainability targets.

Several international reports have highlighted the transformative role of renewables in the energy transition. Gielen et al. [24] examined global decarbonisation scenarios and found that increasing renewable energy penetration could lead to substantial reductions in CO_2 emissions across all sectors, particularly electricity. Their findings were supported by the International Energy Agency (IEA) [25], which asserted that countries with high renewable adoption display lower emissions growth, especially when renewables displace fossil-based generation during peak demand hours.

The economic case for renewables is also strengthening. A report by IRENA [26] indicated that the cost of solar PV and onshore wind has dropped significantly over the past decade, making them competitive with fossil-based generation even without subsidies. As a result, integrating these sources into national grids is not only environmentally favourable but also increasingly cost-effective. Yet, the variable nature of renewables imposes operational constraints on the grid, which must be considered in forecasting models to avoid underestimation or overestimation of emissions.

Several researchers have attempted to capture the emissions impact of renewable deployment. Del Río and Mir-Artigues [27] discussed the importance of policy mechanisms such as feed-in tariffs and renewable portfolio standards in incentivising clean energy adoption. Fischer et al. [28] further argued that grid flexibility and storage technologies are necessary to prevent renewable curtailment and ensure stable grid operation. However, while these studies offer valuable policy insights, they often lack a predictive component that can model the direct emissions outcomes under different renewable scenarios.

Some works have moved towards quantifying the emissions benefits of renewables through data-driven methods. For example, Zhang et al. [29] showed that fluctuations in renewable generation could affect emissions forecasting accuracy, particularly when renewables displace fossil generation inconsistently. Wang et al. [30] echoed this finding, noting that models which fail to incorporate generation mix variability may misrepresent the emissions trajectory. Still, these studies stop short of offering simulation-based forecasting under changing renewable shares.

To bridge this gap, researchers have begun exploring the inclusion of energy mix ratios as model features. Zhou and Chan [31] demonstrated that incorporating the percentage contribution of renewables and fossil fuels improved both interpretability and accuracy in forecasting emissions. Similarly, De Felice and Alessandri [32] integrated exogenous weather variables and energy mix data into hybrid models, successfully improving the prediction of renewable output and its effects on emissions. Yet, most of these models are static in nature and lack the capability to simulate alternative renewable adoption scenarios.

The current study addresses this need by implementing a scenario-based forecasting approach that incorporates renewable share as a dynamic input feature in a deep learning framework. By systematically modifying renewable penetration levels and evaluating the corresponding changes in CO_2 emissions, the model provides a flexible tool for both retrospective analysis and forward-looking policy simulation. This approach not only quantifies the emissions impact of renewable adoption but also allows researchers and policymakers to evaluate hypothetical decarbonisation strategies under real-world grid conditions.

2.5 Identified Gaps and Research Motivation

While a considerable volume of literature exists on electricity demand forecasting, emissions modelling, and renewable energy integration, several critical gaps remain that limit the practical application of existing methods for sustainability-focused energy planning. These deficiencies are especially evident in the siloed treatment of demand and emissions forecasting, the static nature of emissions models, and the limited inclusion of renewable energy dynamics as a variable factor.

A primary shortcoming in many prior studies is the treatment of electricity demand and CO_2 emissions as independent forecasting tasks. Classical models such as ARIMA [2] and early neural network implementations [15, 4] often focused exclusively

on demand-side metrics without incorporating emissions outputs. Even advanced deep learning studies that leverage LSTM architectures [18, 33] typically confine their predictions to either demand or emissions, but not both. This separation fails to account for the inherent relationship between how electricity is produced (via fossil fuels or renewables) and the resulting environmental impact, limiting the usefulness of forecasts for integrated planning.

Another prominent limitation is the static input assumption adopted in many emissions studies. Works like those of Ang and Zhang [1] or Singh and Verma [8] focus on decomposition analysis or statistical extrapolation based on historical emissions drivers, but do not simulate future emissions scenarios under variable conditions. Such models offer insight into past trends but lack the ability to inform proactive decision-making, especially in rapidly evolving power systems. By contrast, dynamic, scenario-based forecasting is increasingly essential for managing decarbonisation pathways, particularly in light of the growing penetration of renewables.

The underutilisation of renewable energy dynamics within predictive models is another critical gap. Although some recent efforts, such as those by Zhang et al. [29] and Zhou and Chan [31], have incorporated generation mix ratios into their frameworks, most existing studies rely on static or lag-based features rather than treating renewable penetration as a scenario-adjustable input. This limitation constrains the model's ability to support "what-if" simulations — a crucial capability when evaluating the policy impact of different clean energy targets.

Moreover, the lack of multi-output forecasting frameworks means that demand-side variables and environmental outcomes are still largely treated in isolation. The complexity of real-world energy systems demands models that can simultaneously predict both demand and emissions under varying operational and policy conditions. Yin et al. [23] demonstrated the promise of multi-task learning for price and demand forecasting, but their approach was not extended to emissions, leaving a gap that this study seeks to address. To fill these gaps, the present research proposes an integrated, multi-output LSTM-based forecasting framework that jointly models electricity demand and CO_2 emissions while incorporating renewable energy share as a controllable scenario variable. This approach allows for both retrospective simulations — evaluating how emissions might have changed with higher renewable deployment — and prospective forecasting — predicting future emissions under planned renewable energy trajectories. Additionally, the model is benchmarked against alternative architectures such as standard RNN and hybrid RBM+NN frameworks, offering comparative insights into sequence learning efficacy and model robustness.

Through this methodological design, the study not only enhances forecasting accuracy but also aligns technical modelling efforts with practical needs in energy policy and climate strategy. It offers a flexible, scenario-driven approach that enables deeper understanding of how grid decarbonisation affects both operational performance and environmental impact — a contribution not currently addressed by the majority of existing forecasting literature.

2.6 Summary of Reviewed Literature

To synthesise the findings from the literature, a comparative summary of key studies is presented in the following table. This summary outlines the forecasting models employed, their primary objectives, regional or sectoral focus, and the major insights derived from each study. It also highlights specific research gaps that continue to hinder the development of integrated forecasting systems, especially those that simultaneously address electricity demand, emissions, and the role of renewable energy.

The review reveals that while substantial progress has been made in model performance—particularly with the adoption of AI and hybrid frameworks—most existing approaches continue to treat electricity demand and CO_2 emissions as distinct forecasting problems. Additionally, few studies have explicitly integrated renewable energy share as a dynamic input feature, leaving a crucial opportunity unexplored. These identified limitations provide a strong rationale for the multi-output LSTM framework developed in this research, which seeks to bridge the gap by linking demand and emissions forecasting under varying renewable penetration scenarios.

Sl. No.	**Author(s)**	**Year**	**Model / Method Used**	**Forecasting Target**	**Data / Region**	**Key Findings**	**Research Gap Identified**
1	Khan, Jamil, Iqbal	2022	ARIMA, ANN	Electricity demand	Pakistan	ANN outperformed ARIMA	No emissions forecasting included
2	Yang, Liu	2021	Index Decomposition	CO_2emissions	China (provinces)	Emissions linked to economic growth	Static, not predictive
3	Wang, Zhang, Liu	2020	LSTM + weather data	Renewable generation	China	LSTM captured nonlinear renewable behavior	Did not link to emissions
4	Chen, Xu, Zhang	2020	LSTM Neural Network	CO_2emissions (coal)	China	High accuracy in emissions prediction	No demand modeling
5	Pereira, Andrade, Costa	2022	ANN, regression	Industrial energy use	Industrial sectors	Regression effective in stable cases	Couldn't capture nonlinearity in emissions
6	Singh, Sharma, Gupta	2021	Temp-based regression	Energy consumption	Weather-sensitive areas	Temperature was predictive	Lacked advanced ML
7	Smyl	2020	Hybrid Exp. Smoothing + RNN	Electricity demand	General data	Hybrid outperformed RNN	No emissions considered
8	Hochreiter, Schmidhuber	1997	LSTM (theoretical)	Time-series theory	—	Introduced memory-based learning	Needs domain-specific application
9	Box, Jenkins	1976	ARIMA (theoretical)	Time-series theory	—	Provided linear baseline model	Limited nonlinear handling
10	Yadav, Patel, Soni	2019	ARIMA + ANN hybrid	Power demand	India	Improved accuracy over solo models	Did not include emissions
11	Chen, Wang, Lin	2021	ARIMA, LSTM	Demand under renewables	East Asia	LSTM beat ARIMA with variability	Emissions excluded
12	Zhou, Wang, Chen	2021	Emission model	CO_2 under renewables	China	Renewables affected emissions significantly	Ignored demand forecasting
13	Bhatia, Gupta	2019	Hybrid AI models	Energy use forecasting	India	AI models enhanced prediction	No emissions linkage
14	Ahmad, Ali, Khan	2022	ML (SVM, ANN, RF)	Demand + emissions	India	SVM, ANN good for nonlinear data	No renewable integration

Table 2.1: Summary of Key Literature on Electricity Demand and CO_2 Emissions Forecasting

Chapter 3

Methodology

This research adopts a structured data science approach to forecast electricity demand and CO_2 emissions within the U.S. electricity sector, with a particular focus on two key regions: California (CAL) and Texas (TEX). To ensure a systematic, replicable, and transparent process, the methodology follows the Cross-Industry Standard Process for Data Mining (CRISP-DM) framework. CRISP-DM remains one of the most widely accepted frameworks for predictive modeling and data mining tasks due to its flexibility, modularity, and compatibility with both statistical and machine learning approaches.

The standard CRISP-DM process includes six phases: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment. However, given the academic nature of this research and its focus on methodological rigor rather than business operations or real-time deployment, certain phases of the framework were adapted or omitted. Specifically, the Business Understanding, Deployment, and Ethical Consideration phases were not included, as the study does not involve commercial decision-making, stakeholder engagement, or direct implementation into operational systems. Instead, the focus of this work is concentrated on three critical phases that directly contribute to the modeling workflow: Data Understanding, Data Preparation, and Modeling. A visual representation of the adapted CRISP-DM framework applied in this study is presented in Figure 3.1.

The Data Understanding phase involves thorough exploration of the dataset, identification of missing values, assessment of variable distributions, and detection of key relationships between electricity demand, generation mix, and emissions. This phase provides the foundational knowledge required to guide appropriate data cleaning and feature engineering strategies.

The Data Preparation phase focuses on preprocessing the dataset to ensure that it is suitable for machine learning models. This includes handling missing data, outlier detection, creation of lag features, generation of rolling statistics, and extraction of time-based attributes such as hour of the day, day of the week, and month of the year. The feature engineering step is particularly critical in time-series forecasting tasks, as it enhances the model's ability to recognize patterns and dependencies within the data. In the Modeling phase, advanced machine learning architectures are employed to perform time-series forecasting. Two primary modeling approaches are adopted in this

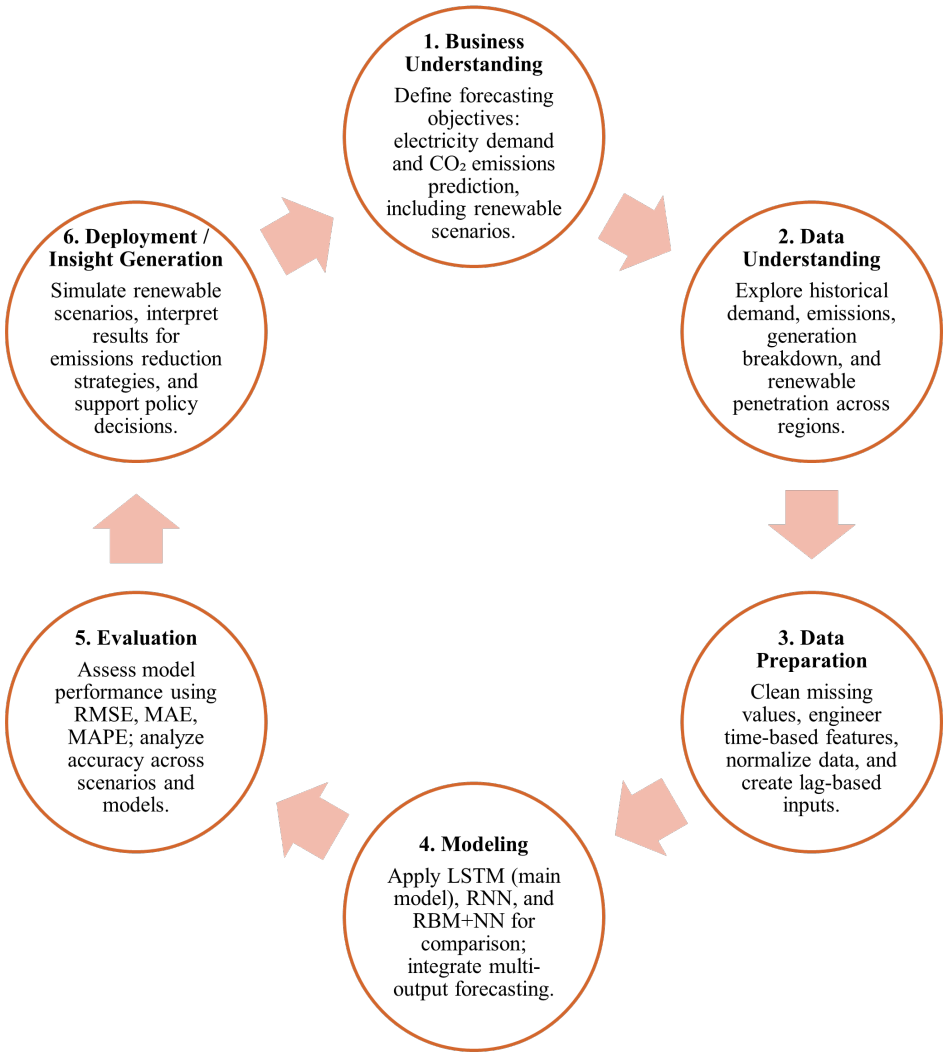

Figure 3.1: Adapted CRISP-DM Framework for Energy Demand and CO_2 Emissions Forecasting

research. The first approach utilizes Recurrent Neural Networks (RNNs), which are well-suited for sequential data due to their ability to retain information across time steps. The second approach explores a hybrid model combining a Restricted Boltzmann Machine (RBM) with a standard Neural Network. The RBM component aids in unsupervised feature learning, allowing the network to capture latent structures in the input data before feeding into the supervised forecasting model. Both architectures are evaluated based on their forecasting performance in predicting electricity demand and associated CO_2 emissions.

This multi-model approach allows for a comparative analysis of different neural architectures, providing insights into which techniques are most effective for the specific context of energy forecasting. Model selection and performance evaluation are guided by well-established metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE), which are discussed in detail in the modeling section.

By systematically following this adapted CRISP-DM framework, the research ensures

methodological transparency while focusing on the core technical aspects that contribute to accurate and reliable forecasting outcomes.

3.1 Data Understanding

Understanding the structure, behavior, and completeness of the dataset is a critical step before proceeding with any predictive modeling. This research employs a rich, high-frequency dataset that captures the dynamics of the U.S. electricity grid, including demand, generation, and CO_2 emissions, across multiple regions over a five-year period. The data understanding phase was essential for evaluating the quality of the dataset, identifying patterns, detecting anomalies, and informing the subsequent steps of data cleaning, feature engineering, and model development.

3.1.1 Data Source and Description

The dataset used in this study was obtained from the Harvard Dataverse, published by Biswas et al. (2023), and contains hourly operational data from July 2018 to June 2023. The raw data was originally compiled from the United States Energy Information Administration's EIA-930 real-time grid monitoring system, which provides detailed records of electricity consumption, generation by source, and associated CO_2 emissions across twelve key grid regions in the United States. These regions include major markets such as California (CAL), Texas (TEX), the Midwest (MIDA, MIDW), and others.

The core focus of this research narrows down to two contrasting regions: California and Texas. These regions were selected due to their differing energy generation strategies and portfolio compositions. California's grid demonstrates a higher penetration of renewables, particularly solar, while Texas remains significantly reliant on natural gas and wind. This selection enables a comparative analysis of how regional generation structures influence both electricity demand patterns and CO_2 emissions behavior.

The compiled dataset comprises more than 525,000 hourly observations and includes a broad range of variables. These variables cover actual and forecasted electricity demand, net generation levels, generation breakdowns by technology (including coal, gas, nuclear, hydro, solar), CO_2 emissions associated with different fuel types, and emission intensity factors. Temporal attributes such as timestamps, hour of the day, day of the week, and month were also incorporated to allow for the capture of seasonality and cyclicality in the data.

3.1.2 Time Frame, Resolution, and Geographic Scope

The dataset spans a continuous five-year period from July 1, 2018, to June 30, 2023, with hourly resolution. This high-frequency time-series structure provides the necessary granularity to detect short-term operational dynamics, daily load cycles, and longer-term seasonal variations. The decision to focus on hourly data, as opposed to daily or monthly aggregation, was driven by the need to capture peak-load events, ramping behaviors, and renewable intermittency, all of which occur at sub-daily scales and are critical for accurate forecasting.

California and Texas serve as ideal candidates for this analysis not only because of their significant contributions to the national grid but also due to their distinct generation mixes and policy environments. California, with its aggressive decarbonization policies and substantial solar capacity, offers insights into the behavior of renewables-dominant grids. Texas, on the other hand, provides a contrasting case where fossil fuels, particularly natural gas, continue to play a central role, alongside a growing wind sector. This regional diversity ensures that the forecasting models developed in this research are tested across varying grid conditions.

3.1.3 Initial Data Exploration and Summary Statistics

The initial exploration of the dataset involved evaluating the range, distribution, and completeness of key operational metrics. Summary statistics for electricity demand, net generation, and CO_2 emissions were computed to assess central tendencies, variability, and outliers. The average electricity demand across all regions was found to be approximately 36,500 megawatt-hours, with significant fluctuations observed between peak and off-peak periods. Net generation values reflected similar variability, influenced by both demand cycles and renewable generation availability. Emissions data exhibited pronounced peaks during periods of high fossil generation, confirming the expected relationship between demand fulfillment and carbon output.

To visually assess these temporal behaviors, weekly averaged electricity demand was plotted over the entire period of study. This line plot, presented as Figure 3.2, highlights the cyclical nature of power consumption, with recurring annual peaks typically observed during summer months. The pattern aligns with seasonal cooling loads and supports the need for time-aware forecasting models.

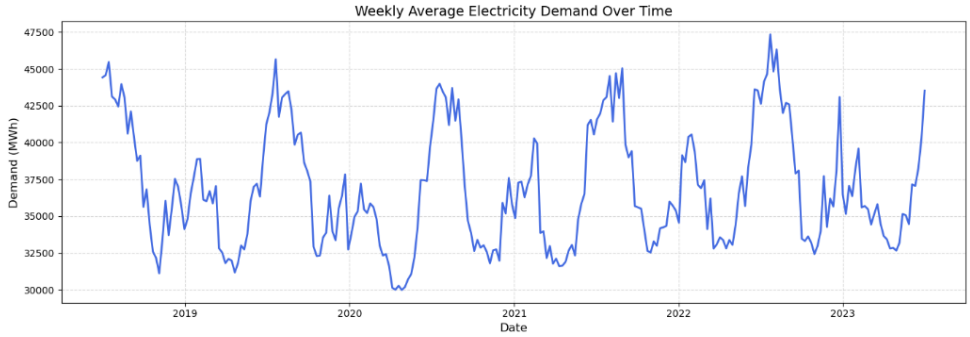

Figure 3.2: Weekly Average Electricity Demand Over Time (2018–2023)

The relationship between electricity demand, net generation, and CO_2 emissions was further explored through a multi-variable line plot, shown as Figure 3.3. This visualization demonstrated that spikes in demand are closely followed by increases in generation and emissions, reinforcing the operational linkages among these key variables.

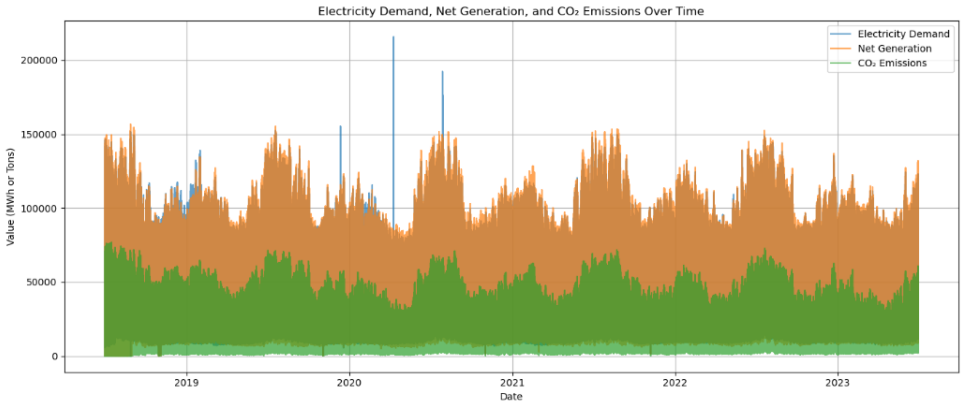

Figure 3.3: Electricity Demand, Net Generation, and CO_2 Emissions Over Time

3.1.4 Energy Mix Patterns and Emissions Contribution

Understanding the composition of the energy mix is essential for interpreting the emissions profile of the power grid. A stacked area chart was generated to illustrate the average monthly contribution of different generation sources, including coal, gas, nuclear, hydro, and solar. The chart, displayed as Figure 3.4, reveals significant seasonal variability in hydro and solar outputs, while coal and gas maintain more stable contributions throughout the year.

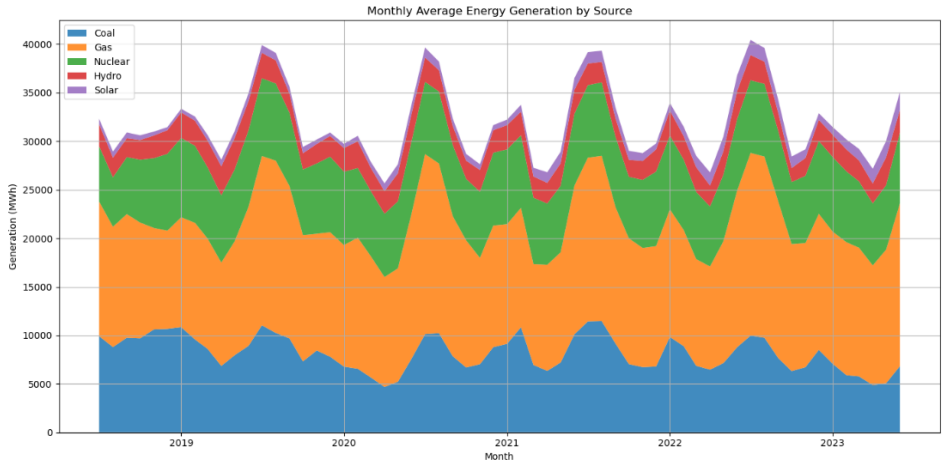

Figure 3.4: Monthly Average Energy Generation by Source (2018–2023)

To assess the proportional contribution of each generation source to total CO_2 emissions, a pie chart was created, as shown in Figure 4. This chart confirmed that coal remains the predominant source of carbon emissions, contributing nearly 59 %, followed by natural gas at approximately 40%. Other generation sources, including renewables, have a negligible direct contribution to emissions, although they indirectly affect emissions by displacing fossil generation.

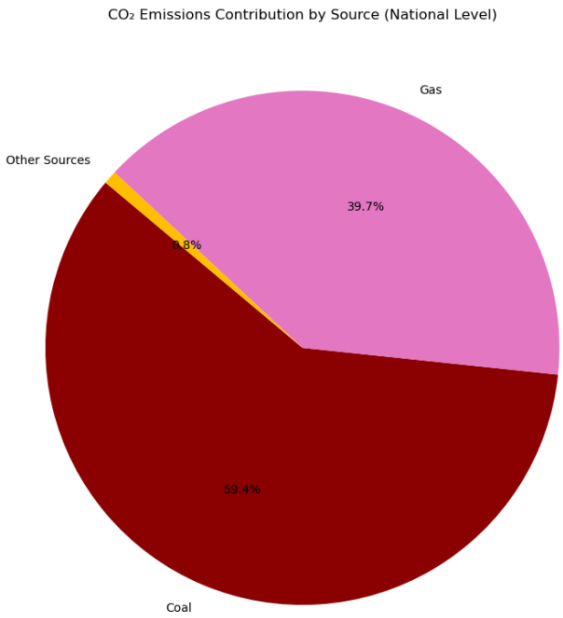

Figure 3.5: CO_2 Emissions Contribution by Source (National Level)

3.1.5 Temporal and Regional Demand Patterns

The temporal demand behavior was analyzed further by decomposing the data across different time units, including hourly, daily, and monthly scales. Violin plots and line charts were employed to illustrate demand distributions across days of the week and hours of the day. These analyses, presented as Figure 3.6 and Figure 3.7, revealed that electricity demand tends to peak on weekdays, particularly during late afternoon and early evening hours, consistent with industrial activity patterns and residential consumption behaviors.

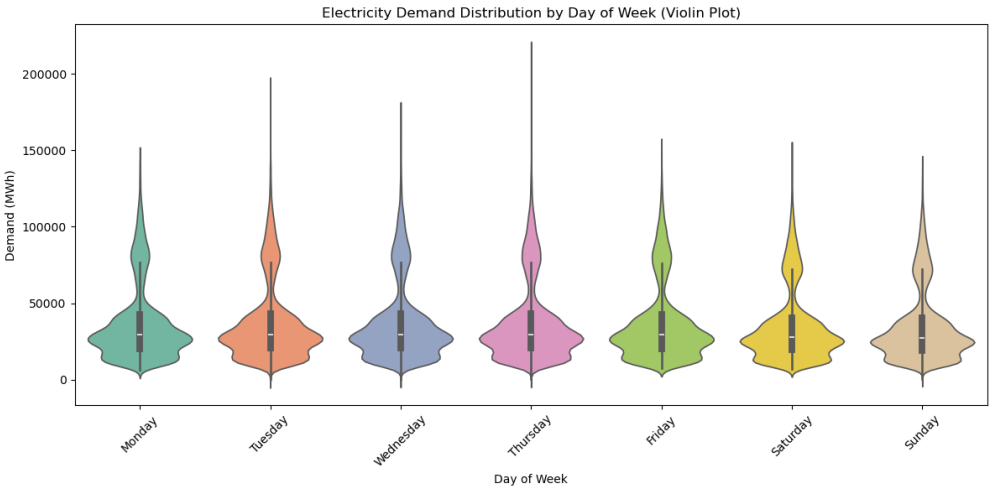

Figure 3.6: Electricity Demand Distribution by Day of the Week

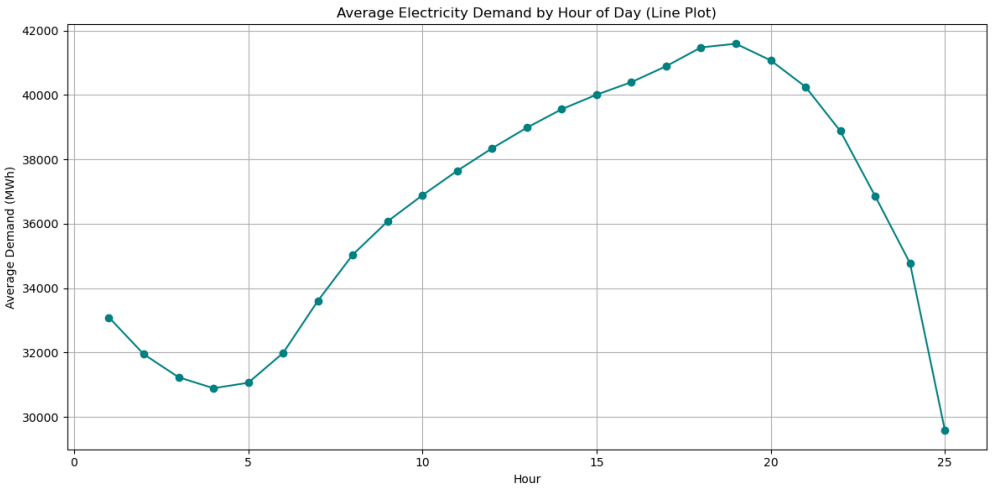

Figure 3.7: Average Electricity Demand by Hour of the Day

Heatmaps were also generated to capture the interaction between hourly demand variations and both day-of-week and seasonal effects. These visualizations, shown as Figure 3.8 and Figure 3.9, demonstrated the presence of clear behavioral patterns that justify the use of lag features and seasonality-aware forecasting models.

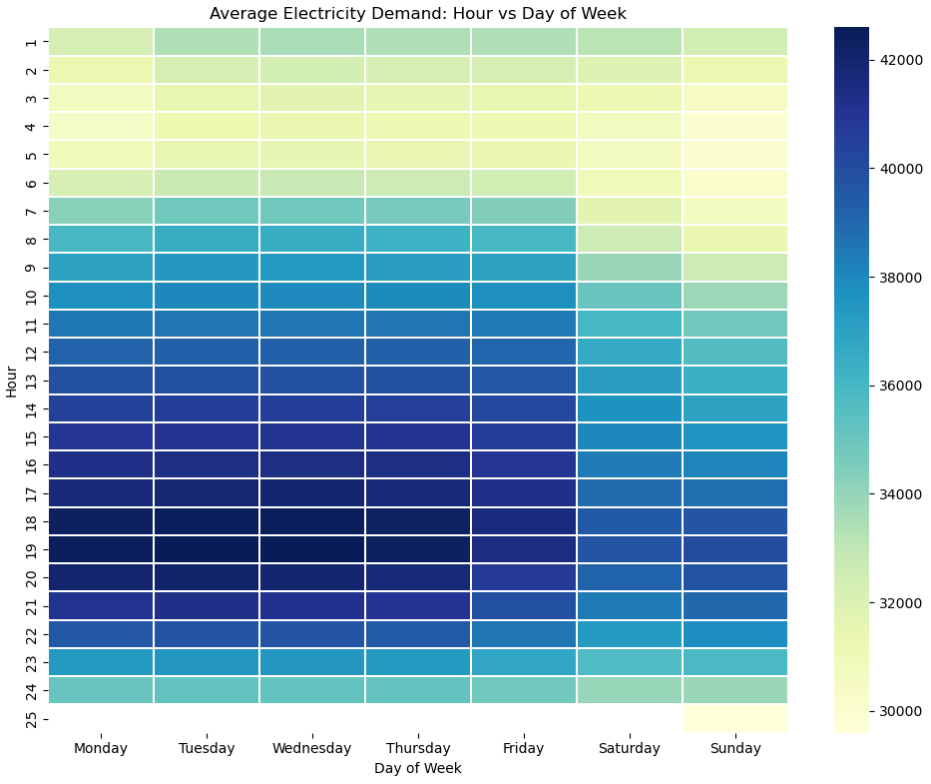

Figure 3.8: Heatmap of Average Electricity Demand by Hour vs. Day of the Week

Figure 3.9: Heatmap of Average Electricity Demand by Hour vs. Month of the Year

In addition to temporal decompositions, regional comparisons were performed to examine how demand behaviors differ between California and Texas. Heatmaps specific to these two regions, displayed as Figure 3.10 and Figure 3.11, confirmed distinct demand cycles and peak load timings, reflecting their different consumption patterns and generation structures.

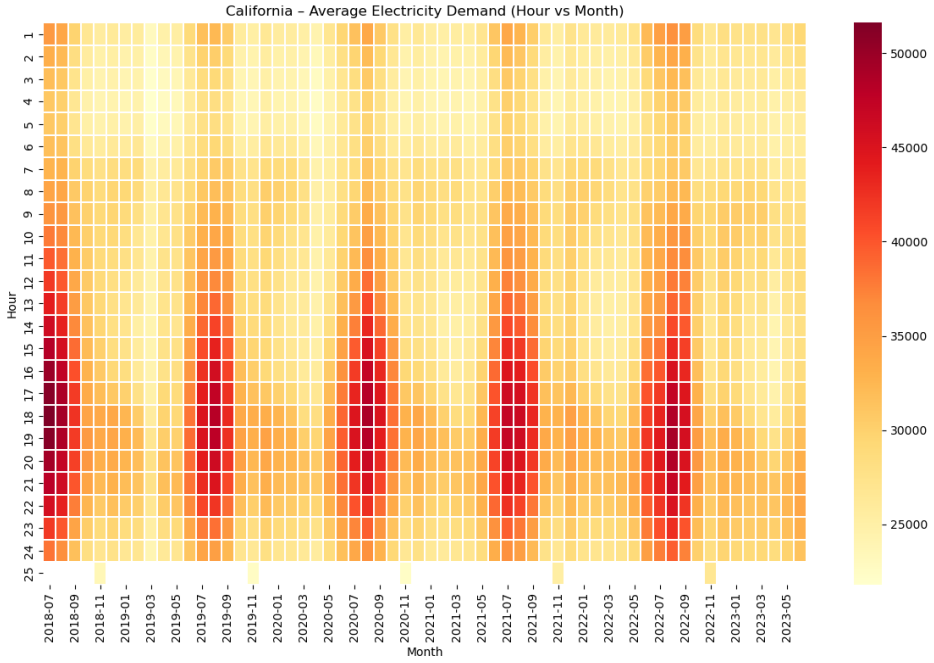

Figure 3.10: California Hourly Demand Pattern by Month

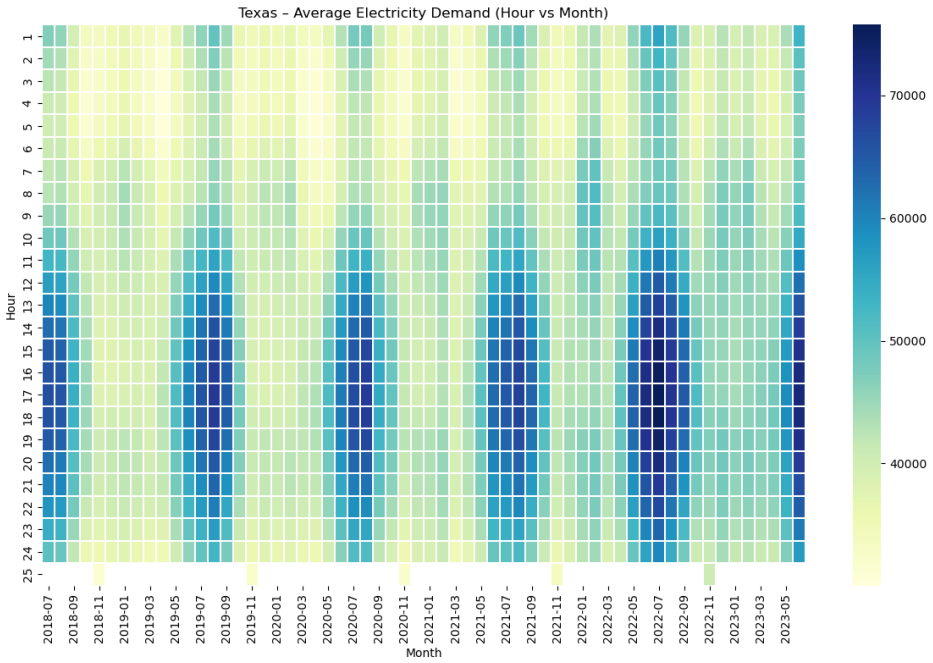

Figure 3.11: Texas Hourly Demand Pattern by Month

3.1.6 Demand and Emissions Comparison Across Regions

A combined bar-line chart was generated to compare the total electricity demand and total CO_2 emissions across all regions. This chart, presented as Figure 3.12, revealed that the Midwest regions (MIDA and MIDW) lead in both demand and emissions, while Texas consistently ranks high due to its large-scale energy production. California, while maintaining substantial demand levels, exhibits relatively lower emissions, reflecting the influence of its higher renewable energy penetration.

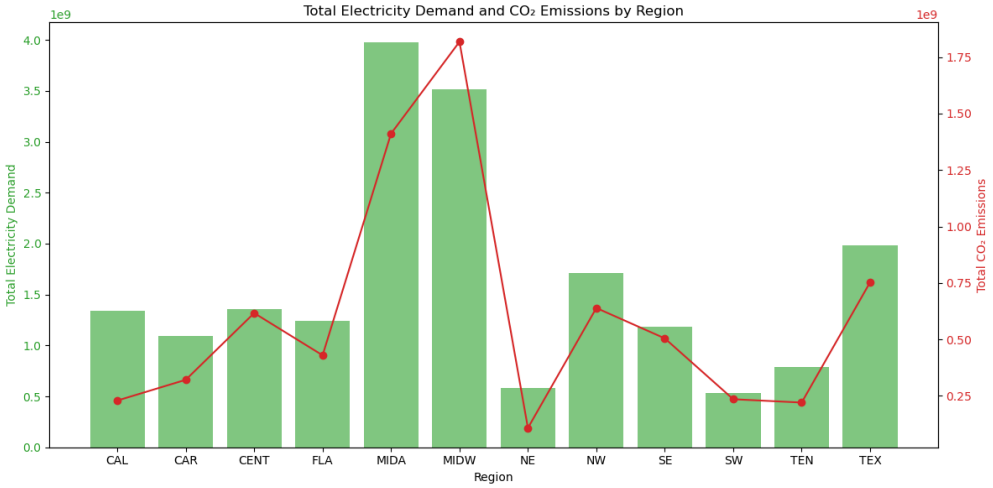

Figure 3.12: Total Electricity Demand and CO_2 Emissions by Region

3.1.7 Renewable Energy vs Emissions Trend Analysis

To gain an exploratory understanding of the relationship between renewable energy penetration and carbon dioxide emissions, the study included a comparative visual-

ization of renewable energy share and CO_2 emissions for both California and Texas. These plots span multiple years of hourly data, highlighting how fluctuations in renewable contribution align with emissions outcomes.

In California, the graph (Figure 3.13) shows a clear inverse relationship: as the share of renewable energy increases, CO_2 emissions tend to decrease, particularly during the spring and summer months when solar output is at its peak. This dynamic illustrates the direct decarbonization potential of higher renewable energy integration in regions with well-established green infrastructure.

In contrast, Texas (Figure 3.14) exhibits a more complex pattern. While there are periods where increased renewable share (primarily wind) corresponds with reduced emissions, the correlation is not as strong or consistent as in California. This is likely due to Texas's heavier reliance on natural gas and the intermittent nature of wind energy. Nonetheless, the data still indicate that boosting renewable contributions has a mitigating effect on emissions, though this effect may be moderated by other operational factors in fossil-heavy grids.These historical visualizations are not forecasting outputs, but rather serve to contextualize the importance of modeling renewable share as a dynamic input feature for predictive emissions modeling.

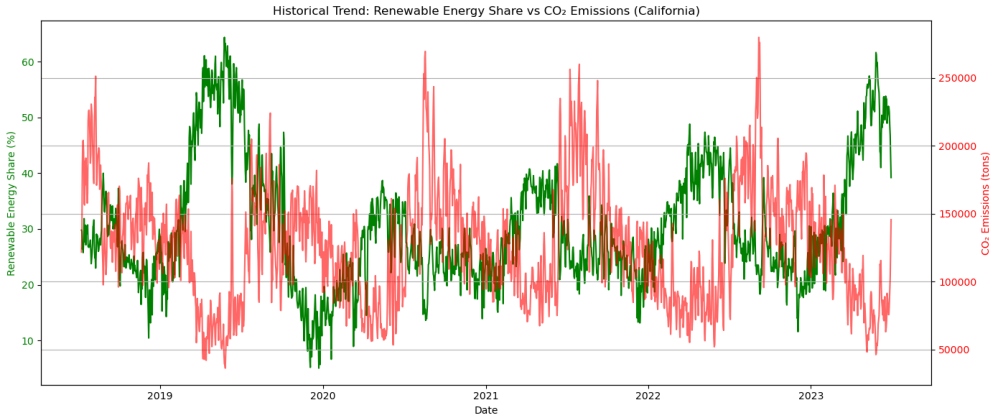

Figure 3.13: Historical Trend – Renewable Energy Share vs CO_2 Emissions (California)

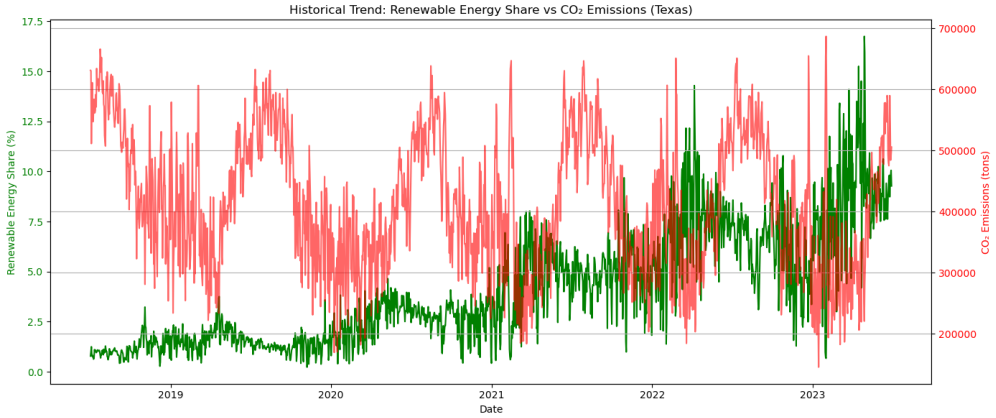

Figure 3.14: Historical Trend – Renewable Energy Share vs CO_2 Emissions (Texas)

3.2 Data Preparation

The preparation of data for this study involved a series of methodical steps aimed at ensuring the quality, continuity, and analytical readiness of the dataset for forecasting electricity demand and CO_2 emissions. Given the time-series nature of the dataset and its high-frequency hourly resolution, particular attention was given to cleaning operations, handling of missing values, feature engineering, and final transformations that support model stability and convergence. The following sections describe the entire data preparation workflow in detail.

3.2.1 Data Cleaning Approach

The original dataset, compiled from multiple regional CSV files sourced from Harvard Dataverse and the EIA-930 grid monitoring system, was first consolidated into a unified dataframe. This merging process facilitated consistent analysis across all twelve U.S. electricity grid regions. After concatenation, a selection of 24 key variables was retained, focusing exclusively on those metrics directly relevant to electricity demand forecasting and emissions estimation. Variables associated with oil-based generation, wind generation, and their corresponding emissions factors were excluded due to high levels of missing data and their limited influence on total generation profiles during the selected timeframe.
Ensuring temporal consistency was a critical component of this cleaning phase. The dataset's timestamp entries were converted to a uniform datetime format, and continuity across the five-year period was verified. This preprocessing step guaranteed that each record accurately reflected its respective hourly observation, maintaining the strict sequence required for effective time-series modeling.

3.2.2 Handling Missing Values

A structured approach was applied to address missing values across the dataset. The strategy for imputation and gap filling was determined based on the operational logic of electricity generation and emissions reporting. Missing values in generation columns were interpreted as non-operation during those hours and were thus filled with zeros. Emission factor gaps were addressed through mean substitution, while missing demand and emissions values were treated using linear interpolation to preserve the sequential integrity of the time series.
The complete strategy for handling missing values is summarized in Table 3.1.

Handling Strategy	**Applied On**	**Method Used**
Zero-fill	Generation columns (e.g., `Coal_Gen`, `Gas_Gen`)	Assumed non-generation (`fillna(0)`)
Mean-fill	Emission factors (`CO2_Factor_Coal`, `CO2_Factor_Gas`)	Replaced with column mean
Linear interpolation	Demand, generation, emissions columns	Time-based linear interpolation

Table 3.1: Summary of Missing Data Handling Strategies

Following these procedures, the cleaned dataset comprised 525,474 hourly observations across 20 variables, with no remaining missing entries. This fully prepared dataset served as the basis for subsequent feature engineering.

3.2.3 Feature Normalization

To support the stability of the forecasting models and promote faster convergence during neural network training, all numeric variables were normalized by rounding their values to two decimal places. This rounding minimized floating-point precision errors without sacrificing meaningful variation in the data. More advanced scaling techniques, such as standardization or min-max normalization, were applied later within the modeling phase, where algorithm-specific scaling requirements were considered.

3.2.4 Time-Based Feature Engineering

Recognizing the importance of temporal dynamics in electricity demand and emissions behavior, a suite of time-based features was engineered from the timestamp data. These included the month of the year, day of the week, binary weekend indicators, day of the year, ISO week numbers, and seasonal groupings. Seasonal categories (winter, spring, summer, autumn) were further processed using one-hot encoding, producing four binary variables that reflect seasonal patterns without introducing ordinal bias into the machine learning models. The summary of these time-based features is provided in Table 3.2.

Feature Name	**Description**	**Purpose**
Month	Month extracted from timestamp (1–12)	Captures seasonal variation
DayOfWeek	Day of the week (0 = Monday, 6 = Sunday)	Differentiates weekday vs. weekend behavior
Is_Weekend	Binary flag indicating weekends	Captures industrial demand differences
DayOfYear	Day count within the year (1–365/366)	Represents long-term seasonal cycles
WeekOfYear	ISO week number (1–52/53)	Captures weekly patterns and behaviors
Season	Categorized as Winter, Spring, Summer, Autumn	Reflects energy usage variation across seasons
Season One-Hot	One-hot encoded columns for each season	Prevents ordinal bias, improves model performance

Table 3.2: Summary of Temporal Feature Engineering for Forecasting

3.2.5 Lag Features and Rolling Statistics

The autocorrelated nature of electricity demand and emissions data makes lag-based features and rolling statistics essential for accurate time-series modeling. Historical demand values from the previous hour, the same hour on the previous day, and the same hour one week earlier were incorporated as lag features. These features allow the models to detect both short-term persistence and weekly seasonality. Rolling mean features were also engineered using three-hour and twenty-four-hour windows to capture smoothed recent averages, enhancing the model's ability to recognize sustained changes and mitigate the effect of outliers or sudden spikes. The key lag and rolling statistical features are presented in Table 3.3.

Feature Name	**Description**
`Demand_Prev_Hour`	Demand from the immediate previous hour
`Demand_Yesterday_Same_Hour`	Demand from the same hour on the previous day
`Demand_Last_Week_Same_Hour`	Demand from the same hour one week earlier
`Rolling_Mean_3H`	3-hour moving average of demand
`Rolling_Mean_24H`	24-hour moving average of demand

Table 3.3: Summary of Lag Features and Rolling Averages Used in Forecasting

These lagged and rolling features complement the inherent memory capabilities of neural architectures like RNNs while improving the learning of non-linear relationships between time steps.

3.2.6 Energy Mix Ratio Features

The emissions output of a power system is directly influenced by its generation portfolio. To account for this, ratio-based features were developed to quantify the share of renewable and fossil-fuel-based generation in total electricity production. The total generation was calculated as the sum of coal, gas, nuclear, hydro, and solar generation outputs. The renewable share was computed as the combined contribution of hydro and solar generation, while the fossil share represented the combined output of coal and gas. The formulas and interpretations of these energy mix ratio features are summarized in Table 3.4.

Feature Name	**Formula**	**Interpretation**
`Total_Gen`	`Coal_Gen + Gas_Gen + Nuclear_Gen + Hydro_Gen + Solar_Gen`	Total electricity generation from all major sources
`Renewable_Pct`	`(Hydro_Gen + Solar_Gen) / Total_Gen × 100`	Percentage share of generation from renewable sources
`Fossil_Pct`	`(Coal_Gen + Gas_Gen) / Total_Gen × 100`	Percentage share of generation from fossil fuels

Table 3.4: Energy Mix Ratio Features and Their Interpretations

These features enhance the model's capacity to recognize how the generation mix affects emissions behavior and overall grid dynamics.

3.2.7 Final Data Transformation

To ensure the dataset's compatibility with machine learning workflows, several final transformations were applied. One-hot encoded categorical variables, such as the seasonal indicators, were converted to binary integers (1 for true, 0 for false). All numeric columns were rounded uniformly to three decimal places to prevent computational inconsistencies.

The dataset was then sorted chronologically by region and timestamp to preserve sequence integrity, an essential requirement for time-series forecasting. Any remaining gaps in the hourly timeline were filled using a forward-backward fill method applied within each region, ensuring that the data maintained complete continuity throughout the entire observation period.

The final processed dataset consisted of 525,060 hourly observations and 36 feature columns, integrating cleaned operational data, engineered temporal features, lag and rolling statistics, and energy mix ratios, providing a robust foundation for the modeling phase.

3.3 Modeling

The core objective of this research is to develop a reliable forecasting model for electricity demand and CO emissions using advanced deep learning techniques. Machine learning models, particularly deep neural networks, have demonstrated significant potential in capturing the non-linear dynamics and temporal dependencies of time-series data. Given the sequential nature of electricity consumption and the influence of past behaviors on future states, recurrent neural architectures were chosen for this task.

Among the various models considered, the Long Short-Term Memory (LSTM) network serves as the primary forecasting model in this research, while Recurrent Neural Networks (RNNs) and a hybrid Restricted Boltzmann Machine (RBM) combined with a feedforward Neural Network (NN) are employed as comparative benchmarks. This section elaborates on the theoretical background, structural design, and learning mechanisms of these models.

3.3.1 Model Selection Rationale

Forecasting electricity demand and emissions involves learning complex temporal patterns and dependencies from historical data. Traditional statistical models such as ARIMA, though effective in capturing linear trends and seasonality, struggle to handle non-linearity and long-term dependencies without substantial manual feature engineering. Neural networks, particularly deep learning architectures, provide a flexible framework for modeling such relationships without strict assumptions about data distributions.

The decision to employ recurrent architectures like RNNs and LSTMs is motivated by their ability to process sequential inputs where the current output depends on previous inputs. However, vanilla RNNs face significant limitations in learning long-term dependencies due to the vanishing gradient problem. To address this, LSTM networks were introduced, incorporating internal memory cells that can retain information over longer sequences. LSTM networks have been successfully applied in various fields such as speech recognition, natural language processing, and financial forecasting, making them well-suited for the energy sector's time-series forecasting challenges.

Additionally, this study includes a hybrid approach that combines an RBM for feature extraction with a standard neural network. The RBM acts as an unsupervised pre-training layer, learning latent feature representations that can enhance the forecasting ability of the subsequent supervised model. This hybrid architecture provides a useful comparison to purely recurrent models.

3.3.2 Long Short-Term Memory (LSTM) Networks

The Long Short-Term Memory (LSTM) network is an advanced type of Recurrent Neural Network (RNN) specifically designed to address the vanishing gradient problem that commonly affects basic RNN architectures. Introduced by Hochreiter and Schmidhuber in 1997, LSTMs incorporate a memory cell mechanism that allows them to retain and regulate information across long sequences. This unique capability makes LSTM networks particularly effective for time-series forecasting tasks where both short-term fluctuations and long-term seasonal patterns play critical roles, such as in the prediction of electricity demand and CO_2 emissions.

Unlike traditional feedforward neural networks, where each input is processed independently, LSTM networks maintain internal cell states that can selectively store, update, or forget information over time. This enables the model to learn which past information is relevant to retain and which can be discarded, overcoming the limitations of basic RNNs where distant dependencies are often lost due to gradient decay during backpropagation.

LSTM Cell Structure and Components

The core building block of an LSTM network is the LSTM cell, which consists of three key gates:

1. Forget Gate: Decides what information to discard from the cell state.
2. Input Gate: Determines what new information should be added to the cell state.
3. Output Gate: Controls the output of the cell and decides how much of the cell state should be passed to the next time step.

Each gate uses a sigmoid activation function to generate values between 0 and 1, which act as weights indicating how much information should be allowed to pass through.

The mathematical operations inside an LSTM cell at time step t are defined as follows:
Forget Gate:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3.1}$$

Input Gate:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{3.2}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{3.3}$$

Update Cell State:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{3.4}$$

Output Gate:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{3.5}$$

$$h_t = o_t * \tanh(C_t) \tag{3.6}$$

Where:

- x_t is the input at time step t.
- h_{t-1} is the previous hidden state.
- C_t is the cell state at time t.
- W_f, W_i, W_C, W_o are weight matrices.
- b_f, b_i, b_C, b_o are bias vectors.
- σ represents the sigmoid activation function.
- tanh represents the hyperbolic tangent activation function.
- $*$ denotes element-wise multiplication.

The cell state C_t functions as the internal memory of the network, capable of carrying relevant information across many time steps without degradation.

Working Mechanism of LSTM

At each time step, the forget gate first decides which pieces of information from the previous cell state should be forgotten. The input gate then identifies which new data should be added to the cell state, and the candidate cell state C_t is computed based on the current input and previous hidden state. These updates ensure that only meaningful information is retained while irrelevant data is discarded. Finally, the output gate determines what information from the updated cell state should be sent to the next layer or next time step.

This selective update mechanism allows the LSTM to effectively learn both short-term dependencies (through its gating structure) and long-term dependencies (via the cell state memory), which is essential for modeling complex seasonal cycles and demand patterns in energy forecasting. Following figure shows the Internal Structure of an LSTM Cell.

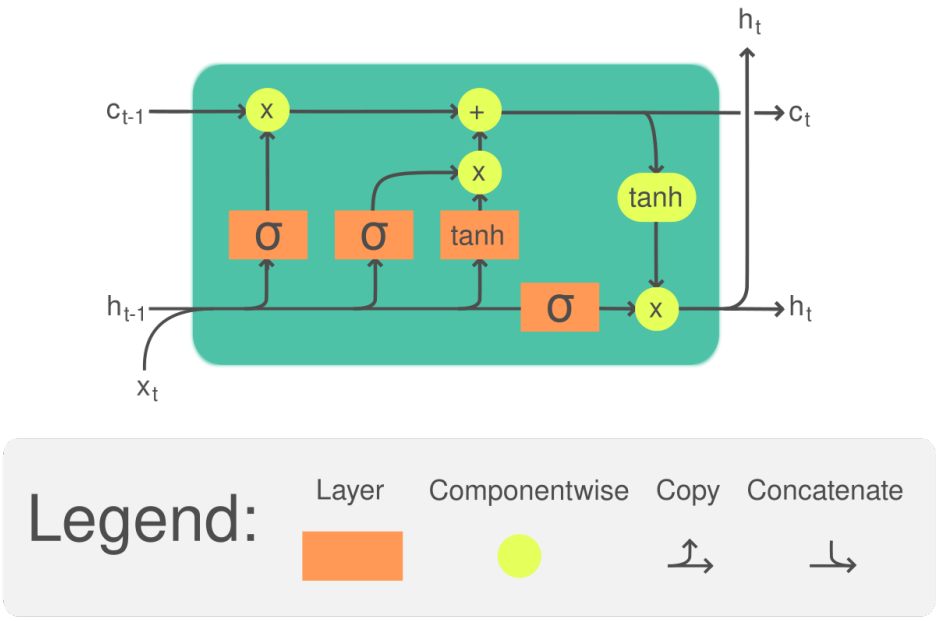

Figure 3.15: Internal Structure of an LSTM Cell Showing Forget Gate, Input Gate, Output Gate, and Cell State Flow

Advantages of LSTM over Basic RNN

LSTM networks are specifically designed to mitigate the vanishing gradient problem, which limits the effectiveness of basic RNNs in handling long sequences. The introduction of memory cells and gating mechanisms enables LSTMs to learn from data spanning longer horizons without significant loss of gradient information during training. This makes LSTMs especially suitable for electricity demand forecasting, where long-term seasonal effects, daily cycles, and behavioral patterns influence energy consumption trends.

In comparison to standard RNNs, LSTMs also provide enhanced flexibility in learning varying time dependencies across different sequences. This capability becomes particularly valuable when forecasting across different regions (such as California and Texas) that exhibit distinct seasonal characteristics and grid behaviors.

3.3.3 Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNNs) represent one of the earliest deep learning architectures specifically designed for modeling sequential and time-series data. Unlike traditional feedforward neural networks where each input is processed independently, RNNs incorporate a feedback mechanism that allows information to persist across time steps. This ability to maintain an internal state or "memory" makes RNNs well-suited for tasks where the current prediction depends not only on the present input but also on previous observations — a key requirement in electricity demand and CO_2 emissions forecasting.

In time-series problems, the sequence of inputs carries essential temporal dependencies.

RNNs capture these dependencies by looping the output of the previous hidden state back into the network, influencing the computation of the current hidden state. This recursive nature distinguishes RNNs from conventional neural networks and allows them to process data sequences of arbitrary length.

Architecture and Working Principle of RNN

At its core, a Recurrent Neural Network (RNN) contains a set of neurons that process sequences by taking the current input x_t and combining it with the hidden state from the previous time step h_{t-1} to produce the current hidden state h_t.
The mathematical formulation of this process can be expressed as:

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{3.7}$$

$$y_t = W_{hy}h_t + b_y \tag{3.8}$$

Where:

- x_t is the input at time step t.
- h_t is the hidden state at time t.
- y_t is the output at time t.
- W_{xh}, W_{hh}, W_{hy} are the weight matrices for input-to-hidden, hidden-to-hidden (recurrent), and hidden-to-output connections, respectively.
- b_h and b_y are bias terms.
- σ is the activation function, typically tanh or ReLU.

The hidden state h_t serves as a memory vector that carries relevant information from previous time steps, enabling the network to make context-aware predictions at each point in the sequence.

Unfolding RNN Through Time

The architecture of an RNN can be visualized by *unfolding* the recurrent loop across time steps. This unrolling process reveals that at each time step, the network essentially becomes a deep neural network with as many layers as there are time steps in the sequence.

The key distinction is that the weights W_{hh} are shared across all time steps, ensuring consistency in learning throughout the sequence. This shared parameter structure allows RNNs to generalize patterns learned at one time step to all others, which is particularly important for sequential data modeling.

Limitations of RNN: Vanishing and Exploding Gradients

Despite their conceptual suitability for sequence learning, standard RNNs are significantly limited by the vanishing gradient problem during backpropagation through time (BPTT). As the gradients of the loss function are propagated backward through many time steps, they can either shrink to near zero (vanish) or grow exponentially

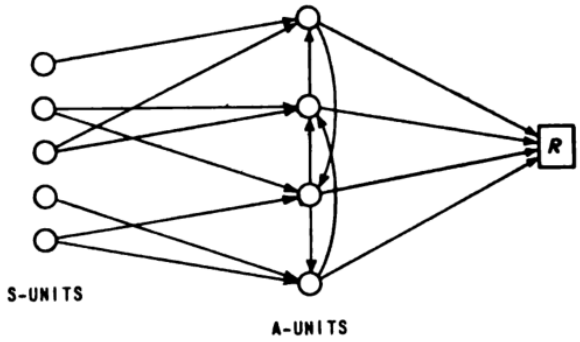

Figure 3.16: Unfolded Structure of a Simple RNN Across Three Time Steps

(explode). In the case of vanishing gradients, the network becomes unable to learn long-range dependencies because the contribution of earlier time steps diminishes exponentially.

This issue arises from the repeated multiplication of the recurrent weight matrices W_{hh} during gradient updates. If these matrices contain values smaller than one, the gradient tends toward zero as the number of time steps increases. Conversely, if the weights are too large, gradients may explode, leading to unstable learning.

These limitations severely restrict the ability of basic RNNs to retain information across long sequences, making them less effective for tasks like electricity demand forecasting where multi-day, weekly, or seasonal dependencies are critical.

Context of RNN Usage in This Study

In this research, the RNN architecture was implemented primarily as a baseline model to evaluate its ability to forecast electricity demand and CO_2 emissions. While capable of modelling short-term dependencies, the RNN's performance on longer sequences was expected to be limited due to its architectural constraints. This justified the choice of LSTM as the primary model, given its superior handling of long-range dependencies through its gated memory structure.

Nevertheless, including RNN as part of the modeling pipeline provides a useful comparative baseline to assess the improvements offered by advanced recurrent architectures like LSTM.

3.3.4 Restricted Boltzmann Machine (RBM) Combined with Neural Network

The Restricted Boltzmann Machine (RBM) is a type of unsupervised probabilistic graphical model that belongs to the family of energy-based models. It is primarily designed for feature extraction, dimensionality reduction, and pre-training by learning latent patterns in the input data. Originally introduced by Geoffrey Hinton and Terry Sejnowski in the mid-1980s, RBMs have gained significant attention for their ability to discover meaningful representations from complex datasets.
An RBM consists of two layers:

- **Visible layer** (v): Represents the input features.
- **Hidden layer** (h): Learns latent representations or features.

Importantly, there are no intra-layer connections within either the visible or hidden layers, which distinguishes the RBM from general Boltzmann Machines. Connections exist only between the visible and hidden layers, making the network "restricted" and simplifying the learning process.

Energy Function and Learning Mechanism

The fundamental concept behind RBMs is the *energy function*, which defines the joint probability distribution over the visible and hidden units. The energy of a particular configuration of the visible and hidden units is given by:

$$E(v, h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_j v_i W_{ij} h_j \tag{3.9}$$

Where:

- v_i = the state of visible unit i
- h_j = the state of hidden unit j
- a_i = bias term for visible unit i
- b_j = bias term for hidden unit j
- W_{ij} = weight between visible unit i and hidden unit j

The **probability of a hidden unit being activated** given the visible units is calculated using the logistic sigmoid function:

$$P(h_j = 1 \mid v) = \sigma\left(b_j + \sum_i v_i W_{ij}\right) \tag{3.10}$$

Similarly, the **probability of a visible unit being activated** given the hidden units is:

$$P(v_i = 1 \mid h) = \sigma\left(a_i + \sum_j h_j W_{ij}\right) \tag{3.11}$$

Here, $\sigma(x) = \frac{1}{1+e^{-x}}$ represents the sigmoid activation function.
RBMs are typically trained using a method known as *Contrastive Divergence (CD)*, which approximates the gradient of the log-likelihood using Gibbs Sampling over a limited number of iterations.

RBM as a Feature Extractor for Neural Networks

In this research, the RBM is utilized not as a standalone predictive model but as a pre-training mechanism for enhancing the feature extraction process before feeding the data into a standard feedforward Neural Network (NN). The key motivation behind this approach is that pre-training with RBMs helps initialize the weights of the neural network in a way that captures meaningful structures in the data, improving

convergence rates and potentially enhancing forecasting accuracy.

The Restricted Boltzmann Machine (RBM) functions as an unsupervised learning model that is capable of capturing latent patterns and hidden relationships within the input features. It does this by learning a probabilistic representation of the input data through the interaction of visible and hidden layers. By doing so, the RBM effectively transforms the raw input features into a set of more abstract and informative representations that capture underlying structures in the data. These representations help reduce noise and redundancy, making them more suitable for downstream predictive modeling.

Once the RBM has extracted these hidden features, they are used as input to a conventional feedforward neural network. This second phase of the hybrid model operates in a supervised learning setting, where the network maps the transformed features to specific forecasting targets—namely, electricity demand and CO_2 emissions. The neural network takes advantage of the RBM's preprocessed representations to learn more accurate mappings, thereby improving forecasting performance. This combination of unsupervised pre-training and supervised learning creates a more robust modeling pipeline that enhances the model's ability to generalize to unseen data, particularly in complex, high-dimensional environments such as energy systems forecasting.

RBM + NN Architecture Overview

In the architecture implemented for this study, the RBM layer serves as the initial component of the model. It is responsible for processing the raw input features and transforming them into a set of learned hidden representations. These hidden representations are generated by identifying latent structures in the data, allowing the model to distill essential patterns from potentially noisy or redundant information. The RBM acts as a feature extractor that performs unsupervised learning, meaning it does not rely on labeled output during this stage but instead focuses on modeling the probability distribution of the inputs.

Once the RBM has generated these feature representations, they are fed into the subsequent stages of the model, which consist of a traditional feedforward neural network architecture. This part of the model is made up of one or more dense (fully connected) layers, which process the extracted features through supervised learning. By combining the RBM's unsupervised feature learning with the neural network's supervised prediction capabilities, the overall architecture becomes well-suited to handling complex, high-dimensional data such as that found in energy systems. This hybrid approach aims to enhance the model's generalization ability, thereby improving its forecasting performance on electricity demand and CO_2 emissions across diverse conditions.

Role of RBM + NN in This Study

In this research, the RBM + NN hybrid model serves as one of the benchmark models alongside RNN and LSTM architectures. While the LSTM remains the primary choice due to its superior handling of sequential dependencies, the RBM + NN architecture provides a useful point of comparison to evaluate the benefit of deep feature extraction through unsupervised pre-training.

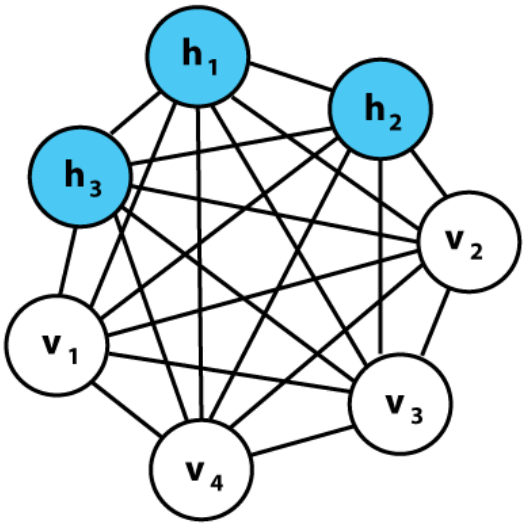

Figure 3.17: Conceptual Diagram of an RBM Combined with a Feedforward Neural Network

3.3.5 Model Training Process

The success of deep learning models in time-series forecasting largely depends on an effective training strategy that enables the networks to learn optimal representations from the data without overfitting or underfitting. In this research, all three modeling approaches — Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), and Restricted Boltzmann Machine (RBM) combined with Neural Network — followed a structured and consistent training process to ensure fairness in comparison and to achieve reliable forecasting outcomes.

The overall training pipeline consisted of several key components, including loss function selection, choice of optimization algorithm, data partitioning strategy, and hyperparameter tuning. This section details the training methodology adopted for the models.

Loss Function

The primary objective of the forecasting models is to minimize the error between the predicted outputs and the actual observed values. For regression-based tasks such as electricity demand and CO_2 emissions prediction, the **Mean Squared Error (MSE)** was used as the main loss function. The MSE penalizes larger errors more heavily, encouraging the models to focus on accurate predictions across the entire range of output values.

The mathematical formulation of the Mean Squared Error is given by:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.12}$$

Where:

- n is the number of observations.

- y_i is the actual observed value.
- $\hat{y}_i$ is the model's predicted value.

In some instances, additional evaluation metrics such as Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) were calculated during the validation phase (these are discussed in the Results and Analysis section).

Optimization Algorithm

The training of neural networks involves iterative adjustment of model weights to minimize the loss function. In this study, the Adam optimizer (Adaptive Moment Estimation) was selected due to its proven efficiency in training deep learning models and its ability to adapt learning rates for individual parameters dynamically.

The Adam optimizer combines the benefits of two traditional stochastic gradient descent extensions: momentum and RMSProp. It maintains exponentially decaying averages of past gradients (first moment) and squared gradients (second moment), which help stabilize the weight updates.

The parameter update rule for Adam is defined as:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \cdot \hat{m}_t \tag{3.13}$$

Where:

- θ: network weights,
- η: learning rate,
- $\hat{m}_t$: first moment (gradient mean),
- $\hat{v}_t$: second moment (gradient variance),
- ϵ: small value for numerical stability.

Batch Size and Epochs

Training was conducted using mini-batch gradient descent, where the dataset was divided into smaller batches. A batch size of 32 was used as a baseline, although alternate batch sizes were experimented with during tuning. The number of epochs — defined as the number of complete passes through the entire training dataset — was set based on early stopping criteria to prevent overfitting. An initial upper limit of 100 epochs was defined, with training termination triggered if the validation loss did not improve over a specified number of consecutive epochs (patience parameter).

Data Splitting and Validation Strategy

To preserve the temporal structure inherent in time-series data, a chronological data splitting technique was employed rather than randomized shuffling. This method ensures the sequence of data points remains intact, which is critical for time-series forecasting where the order of events directly affects future predictions. The dataset was

divided into three distinct sequential subsets: a training set, a validation set, and a testing set. The training set was used to train the model and learn the underlying patterns in the historical data. The validation set was used during training to tune hyperparameters and to implement early stopping to prevent overfitting. Finally, the testing set was reserved for final performance evaluation after all model tuning was completed. This approach of forward-in-time data partitioning mirrors real-world deployment scenarios, where models must forecast future values based solely on past information without access to future outcomes during training.

Hyperparameter Tuning

Hyperparameter tuning played a vital role in optimizing the performance of the forecasting models. Parameters such as the number of neurons in hidden layers, learning rate, batch size, and dropout rates were carefully adjusted using a combination of manual experimentation and grid search strategies. For the LSTM and RNN models, key hyperparameters included the number of recurrent units, which were typically tested within the range of 50 to 200. Dropout rates were varied between 0.1 and 0.5 to determine the optimal level of regularization, while learning rates were explored in the range of 0.0005 to 0.01 to find the most stable and efficient convergence. For the RBM + NN model, the tuning focused on the number of hidden units in the RBM layer, the learning rate used in the contrastive divergence algorithm, and the number of fully connected layers in the downstream neural network. The optimal configurations were identified based on validation performance, ensuring a balance between model complexity and generalization capability.

Regularization Techniques

To mitigate the risk of overfitting, especially in deeper architectures like LSTM, dropout layers were incorporated into the model designs. Dropout randomly deactivates a proportion of neurons during each training iteration, forcing the network to develop redundant pathways and enhancing generalization. In addition to dropout, L2 regularization (weight decay) was applied to penalize large weights, further promoting model robustness.

Chapter 4

Results and Analysis

This chapter presents the evaluation and comparative analysis of the forecasting models developed for electricity demand and CO_2 emissions prediction. The performance of the Long Short-Term Memory (LSTM) network, chosen as the primary model, is assessed alongside two benchmark models: the Recurrent Neural Network (RNN) and the hybrid Restricted Boltzmann Machine combined with a Neural Network (RBM + NN). The evaluation process focuses on measuring the predictive accuracy of these models using standard error metrics, and on analyzing their ability to capture the temporal patterns inherent in the electricity sector data. In addition, the forecasting outputs are visualized to provide a clearer interpretation of how well each model tracks actual demand and emissions across different regions. This chapter also discusses future forecasting results, where applicable, and highlights key observations regarding regional energy behaviors and generation-emissions dynamics.

4.1 Evaluation Framework and Metrics

To evaluate the forecasting performance of the developed models, three widely recognized statistical error metrics were employed: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). These metrics offer complementary insights into the models' predictive accuracy, allowing for both absolute and relative performance evaluation. By leveraging these metrics, the study ensures a robust assessment of the model outputs across varying scales and magnitudes of electricity demand and CO_2 emissions.

The selection of these metrics was based on their ability to handle the non-stationary and seasonal nature of energy-related time series data. Each metric captures different aspects of forecast accuracy: while MAE provides an average measure of error magnitude, RMSE penalizes larger deviations more heavily, and MAPE normalizes the errors into percentages, allowing for intuitive interpretation across regions and time periods.

4.1.1 Mathematical Formulation of Evaluation Metrics

The mathematical definitions of the selected evaluation metrics used in this study are as follows:

Mean Absolute Error (MAE)

The MAE measures the average magnitude of forecast errors, providing a linear score that gives equal weight to all individual differences.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{4.1}$$

Where:

- y_i is the actual value,
- $\hat{y}_i$ is the predicted value,
- n is the total number of observations.

Root Mean Squared Error (RMSE)

The RMSE evaluates the square root of the average squared differences between predicted and actual values. It penalizes larger errors more than smaller ones, making it sensitive to outliers.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{4.2}$$

Mean Absolute Percentage Error (MAPE)

MAPE expresses forecasting errors as a percentage of the actual values, allowing for scale-independent interpretation.

$$\text{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{4.3}$$

4.1.2 Rationale for Metric Selection

The use of MAE, RMSE, and MAPE provides a comprehensive evaluation of model performance across different forecasting horizons and regional contexts. RMSE, with its squared component, emphasizes larger errors and highlights instances where the model significantly deviates from actual observations. In contrast, MAE offers a straightforward and interpretable measure of average error magnitude without disproportionately penalizing outliers. MAPE complements these by offering a scale-independent error percentage, facilitating comparisons across regions with varying levels of demand and emissions.

By employing these three metrics together, the evaluation ensures balanced insight into both absolute and relative forecasting performance, which is critical for supporting policy decisions related to electricity demand management and emissions reduction.

4.2 LSTM Model Results and Analysis

The Long Short-Term Memory (LSTM) model served as the primary forecasting approach in this study due to its proven capability in handling long-range temporal dependencies and nonlinear relationships inherent in energy time series data. The LSTM model was trained on the prepared dataset, which included electricity demand, CO_2 emissions, seasonal indicators, lag-based features, and crucially, the renewable energy contribution percentage (Renewable-Pct). This section presents the performance of the LSTM model at both the national and regional levels, including California and Texas, followed by scenario-based simulations that demonstrate the impact of increased renewable energy on emissions reduction.

4.2.1 National-Level Forecasting Performance

The LSTM model was first evaluated on the aggregated national dataset to forecast daily electricity demand and CO_2 emissions. The model exhibited strong predictive capability, accurately capturing the seasonal fluctuations and non-linear dynamics between demand patterns and emissions behavior.

The achieved forecasting performance metrics are presented in Table 4.1.

Table 4.1: Performance Metrics for Forecasting Models

Target Variable	**RMSE**	**MAE**	**MAPE**
Electricity Demand	6581.10	5626.34	1.24%
CO_2 Emissions	14080.39	11438.30	7.80%

The results in Table 4.2.1 demonstrate that the LSTM model achieved a low Mean Absolute Percentage Error (MAPE) of 1.24% for electricity demand, indicating a high level of accuracy in predicting national consumption trends. The Root Mean Squared Error (RMSE) of 6581.10 and MAE of 5626.34 confirm that the absolute deviation between predicted and actual demand remains relatively small even at the national aggregation level.

In contrast, the CO_2 emissions forecasting yielded a MAPE of 7.80%, with an RMSE of 14080.39 and MAE of 11438.30. While still within acceptable forecasting ranges for emissions modeling, these higher values reflect the inherent complexity and variability in emissions generation, which is influenced not only by total electricity demand but also by fluctuations in generation mix, fuel switching, and operational behaviors of power plants.

The performance gap between demand and emissions predictions is consistent with prior literature, where emissions forecasting typically involves greater uncertainty due to unaccounted external factors such as unexpected outages, energy imports/exports, and weather-induced variability affecting renewables.

The national forecast curve plotted against actual data, shown in Figure 4.1 and Figure 4.2, further illustrates the model's capacity to effectively track key patterns over time.

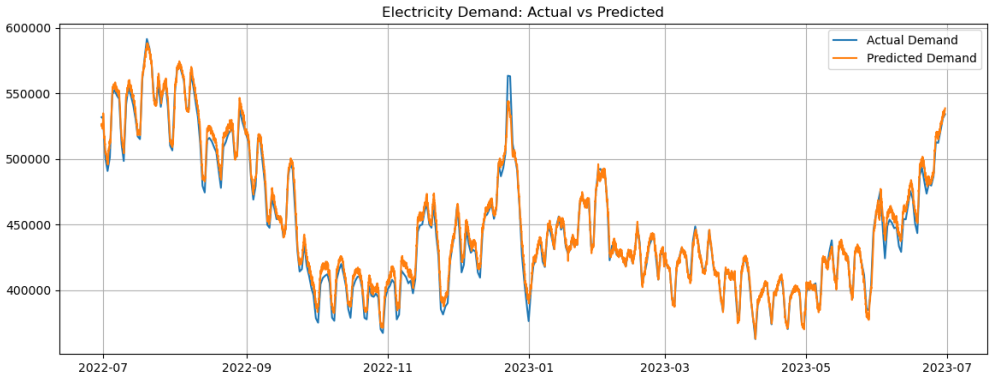

Figure 4.1: Forecasted vs. Actual Electricity Demand (National Level)

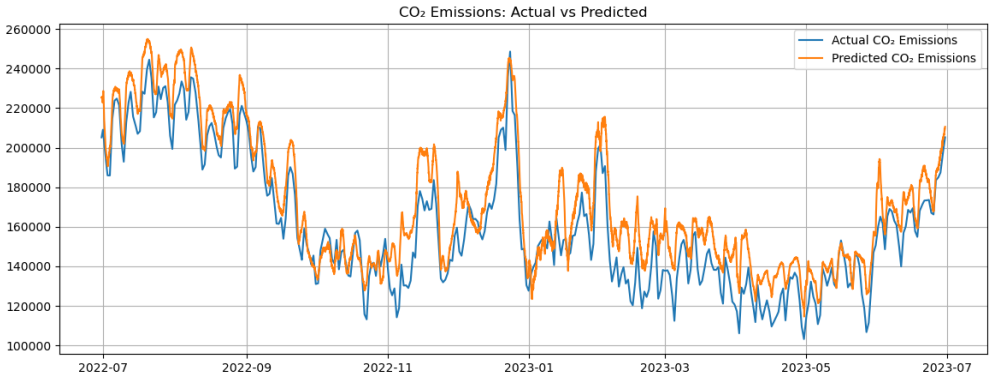

Figure 4.2: Forecasted vs. Actual CO_2 Emissions (National Level)

The close alignment observed in the electricity demand plot confirms the suitability of LSTM for long-range time series forecasting in energy systems, while the emissions forecast, although slightly less precise, still provides actionable insights for carbon management planning.

4.2.2 Regional Forecasting Performance: California and Texas

In addition to the national-level analysis, the forecasting framework was applied to two prominent regional datasets—California (CAL) and Texas (TEX)—to evaluate the adaptability and robustness of the LSTM model under varying energy system conditions. These regions were deliberately selected due to their contrasting energy generation profiles and demand characteristics. California, known for its aggressive renewable energy policies and high penetration of solar and hydroelectric power, exhibits relatively stable electricity demand and CO_2 emissions patterns. This makes it a representative case for renewable-integrated systems. In contrast, Texas maintains a more fossil-fuel-intensive generation mix, with a strong reliance on natural gas and significant contributions from wind. Coupled with its high industrial demand base, Texas presents a more dynamic and volatile consumption profile.

The forecasted vs. actual values for electricity demand and CO_2 emissions in California and Texas are presented in Figure 4.3 , Figure 4.4, Figure 4.5 and Figure 4.6 respectively.

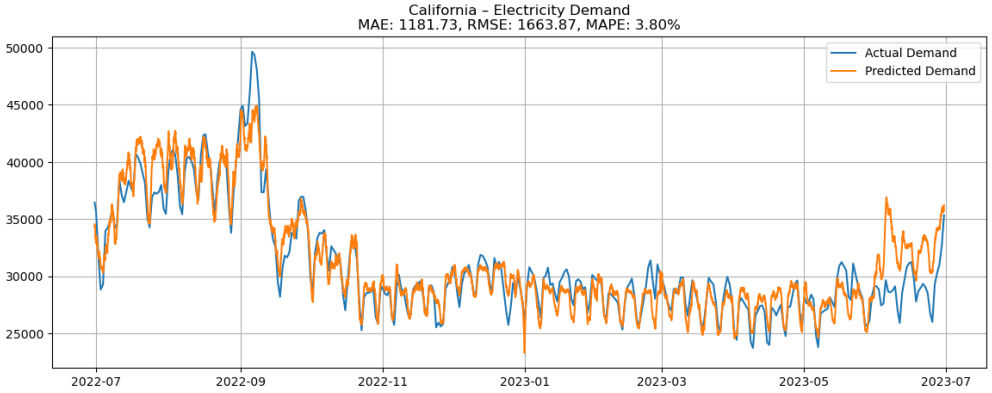

Figure 4.3: Actual vs. Predicted Electricity Demand - California

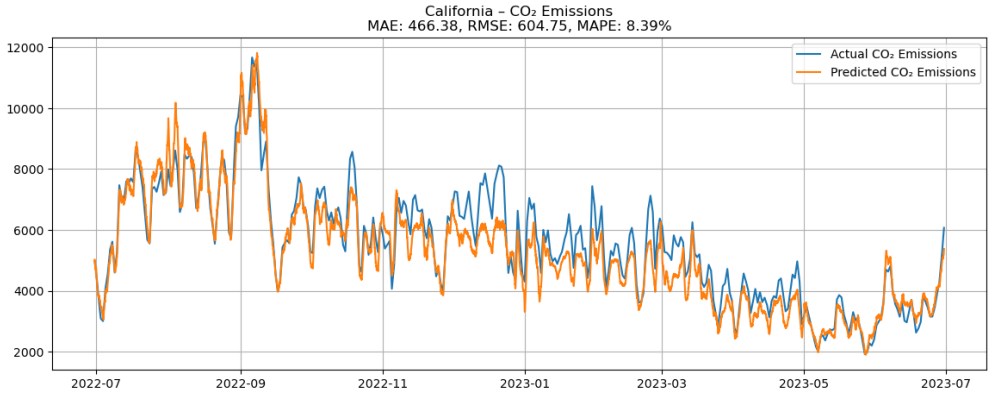

Figure 4.4: Actual vs. Predicted CO_2 emission - California

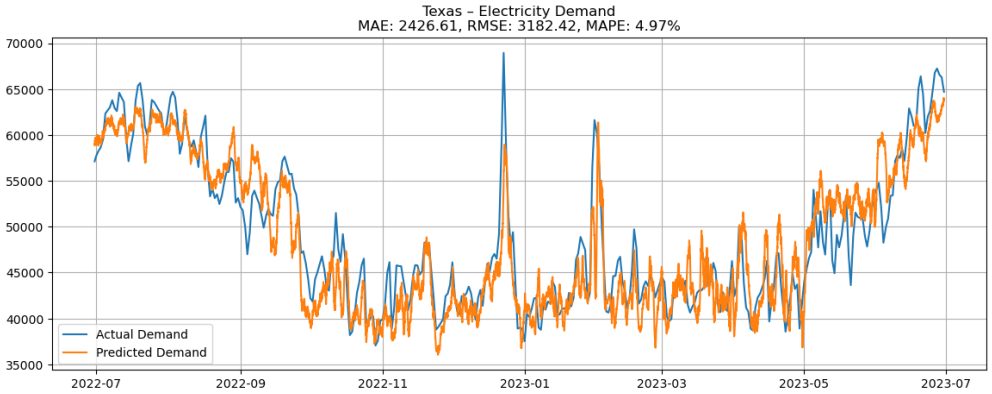

Figure 4.5: Actual vs. Predicted Electricity Demand - Texas

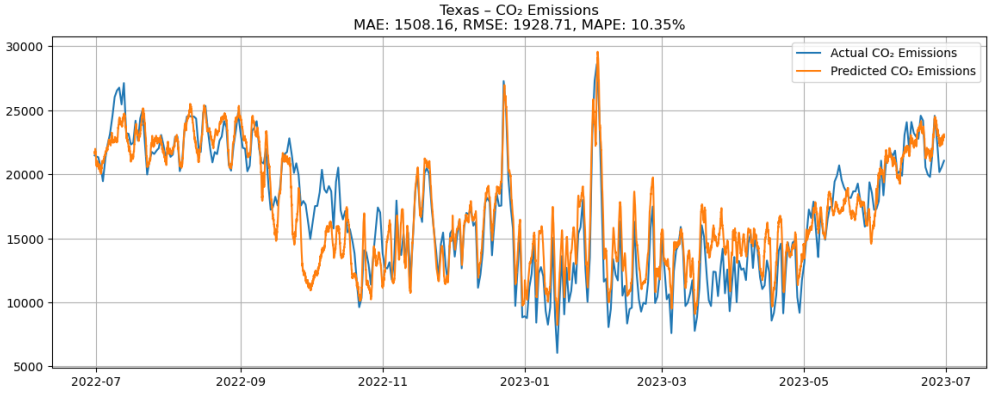

Figure 4.6: Actual vs. Predicted CO_2 emission - Texas

The visual alignment between the actual and predicted curves in both regions highlights the LSTM model's capability to effectively capture seasonal patterns, weekly cycles, and long-term behavioral trends in demand and emissions. The occasional under- or over-estimations observed at peak points, especially in Texas, suggest the influence of factors such as unpredicted load spikes or generation outages, which are not fully captured by the historical features alone.

The LSTM model was tasked with predicting both electricity demand and CO_2 emissions for each of these regions. As summarized in Table 4.2, the model demonstrated high accuracy across both regional contexts, effectively adapting to the unique temporal and structural patterns inherent to each system. In California, the model captured the relatively smooth trends in demand and emissions, while in Texas, it successfully managed more abrupt fluctuations driven by variable fossil generation and load spikes. These results validate the generalizability of the LSTM architecture in modeling energy systems with differing levels of complexity and underscore its suitability for region-specific forecasting applications.

Region	**Metric**	**Demand (RMSE)**	**Demand (MAE)**	**Demand (MAPE)**	**CO_2 (RMSE)**	**CO_2 (MAE)**	**CO_2 (MAPE)**
California	LSTM Forecast	1780.21	1252.87	4.04%	586.74	455.54	8.27%
Texas	LSTM Forecast	3227.03	2514.71	5.13%	1841.79	1462.45	9.99%

Table 4.2: Regional Forecasting Accuracy of LSTM Model (California and Texas)

4.2.3 Analysis of Regional Forecast Results

The regional differences in forecasting accuracy observed between California and Texas reflect the underlying structural and operational characteristics of their respective electricity grids. California's grid, with its substantial share of renewable energy sources—particularly solar and hydro—exhibits smoother, more predictable demand-emission dynamics. This stability, combined with policy-driven decarbonization efforts and relatively flexible grid infrastructure, likely contributes to the model's ability to

learn and generalize well. As a result, the LSTM model was better able to capture the temporal patterns and relationships between demand and CO_2 emissions in California, leading to lower forecasting errors.

Conversely, Texas presents a more complex forecasting environment due to its heavy reliance on fossil fuel-based generation, particularly natural gas, and its high industrial energy consumption. These factors introduce sharp, nonlinear variations in both electricity demand and emissions, especially during periods of peak industrial activity or unexpected generation shifts. The variability in fossil generation also causes CO_2 emissions to respond abruptly to demand changes, making it more difficult for time-series models to consistently predict outcomes during such extreme conditions. Despite these challenges, the LSTM model maintained reliable performance in Texas, demonstrating a MAPE below 10% for CO_2 emissions and within acceptable ranges for electricity demand. This reinforces the model's robustness and adaptability to diverse regional conditions, confirming its practical utility for forecasting tasks in both renewables-dominant and fossil-intensive energy systems.

The performance of the LSTM model across both California and Texas confirmed its robustness in handling time-series energy forecasting under varied grid dynamics. In California, the model exhibited particularly strong predictive accuracy, achieving a Mean Absolute Percentage Error (MAPE) of 4.04% for electricity demand and 8.27% for CO_2 emissions. This high accuracy can be attributed to the relative stability of California's generation profile, marked by substantial renewable energy penetration, which likely facilitated more consistent temporal patterns and improved learning of demand-emissions relationships. Corresponding Root Mean Squared Error (RMSE) values—1780.21 for demand and 586.74 for emissions—further demonstrate the model's precision, indicating limited occurrence of large forecasting deviations.

In contrast, forecasting performance in Texas revealed slightly higher errors. The LSTM model recorded a MAPE of 5.13% for electricity demand and 9.99% for emissions, with RMSE values reaching 3227.03 and 1841.79 respectively. These elevated error margins are reflective of Texas's more volatile energy environment, driven by a heavy reliance on fossil fuels, large industrial loads, and less consistent renewable input. The inherent fluctuations in demand and emissions patterns posed additional complexity for the model, yet the LSTM still maintained an acceptable level of predictive accuracy, underscoring its applicability across grids with diverse operational characteristics.

4.2.4 Scenario-Based Simulation of Renewable Energy Impact on CO_2 Emissions

In addition to the primary forecasting objectives, this study incorporates a simulation-based analysis to assess how increased renewable energy integration affects carbon dioxide (CO_2) emissions at the regional level. This component of the research was motivated by the need to bridge predictive modeling with environmental planning. While previous work often treated demand and emissions separately, or modeled them under static input conditions, this study introduces a dynamic simulation framework using a trained LSTM model to explore both historical and projected emission outcomes

under varying renewable energy penetration levels.

Simulation Framework and Assumptions

The simulation framework employed in this study was designed to evaluate the potential environmental impact of increased renewable energy penetration using the LSTM model trained on historical electricity and emissions data. Two distinct forecasting simulations were developed to provide a comprehensive understanding of the decarbonization potential under different temporal contexts: a historical re-simulation and a future projection. Both simulations aimed to examine how incremental increases in renewable energy contributions might reduce CO_2 emissions, assuming other operational variables remain constant.

In the historical re-simulation, the model revisited the most recent 365-day period from the original dataset, applying hypothetical adjustments to the Renewable-Pct and Fossil-Pct features. These modifications represented idealized increases of +10%, +20%, and +30% in renewable share, with a corresponding linear decrease in fossil fuel-based generation. Crucially, all other input variables—such as electricity demand, seasonal characteristics, and time-based attributes—were held constant to isolate the impact of the energy mix shift on emissions. This approach allowed for a realistic estimation of how emissions would have behaved if a higher proportion of demand had been met by renewables in the recent past, using the same demand patterns and grid structure.

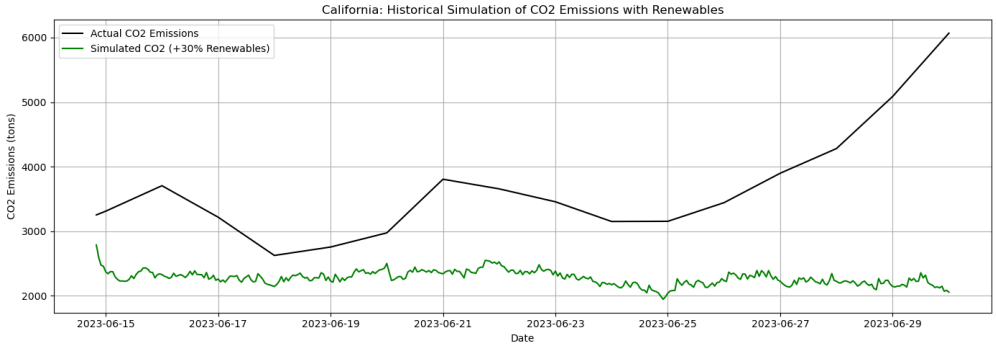

Figure 4.7: California – Historical Simulation of CO_2 Emissions with +30% Renewables

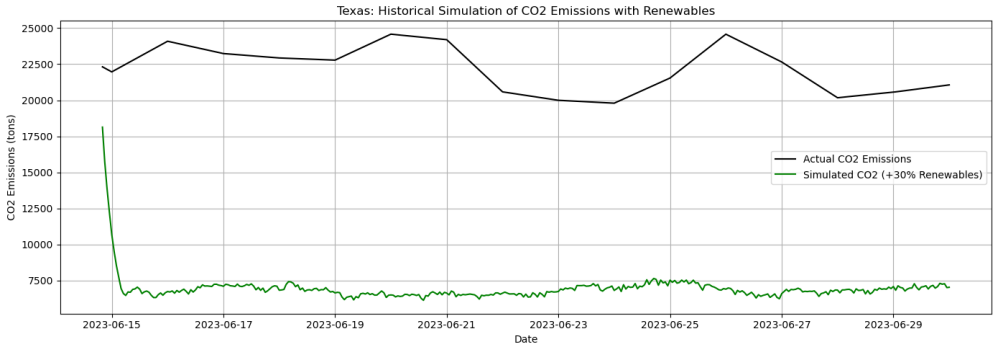

Figure 4.8: Texas – Historical Simulation of CO_2 Emissions with +30% Renewables

The future projection simulation, on the other hand, generated synthetic data to rep-

resent a forward-looking, one-year period. This data included realistic combinations of future dates, weekdays, seasons, and other temporal features, consistent with expected operational timelines. The adjusted renewable and fossil percentages were then incorporated into this forward dataset to evaluate how emissions might respond under altered generation mix scenarios. The trained LSTM model, having learned from historical trends and correlations, was used to forecast daily emissions outcomes over this projected period. Unlike the historical simulation, which preserved the original demand signals, the future projection operates in a predictive mode, extrapolating patterns based on model-learned dynamics.

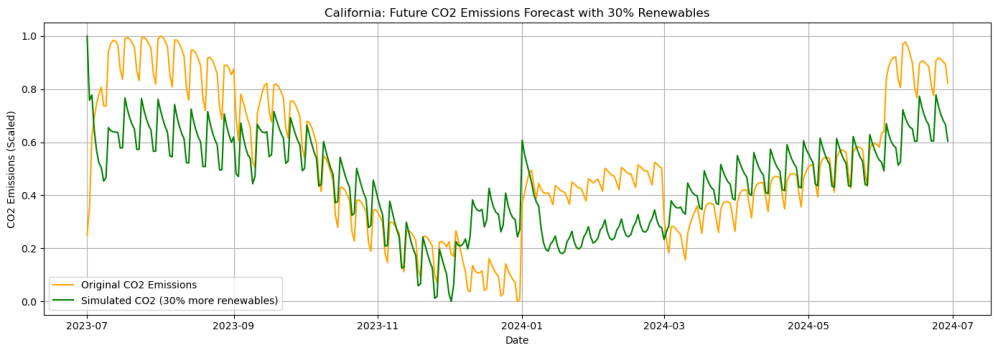

Figure 4.9: California – Future Forecast of CO_2 Emissions with +30% Renewables

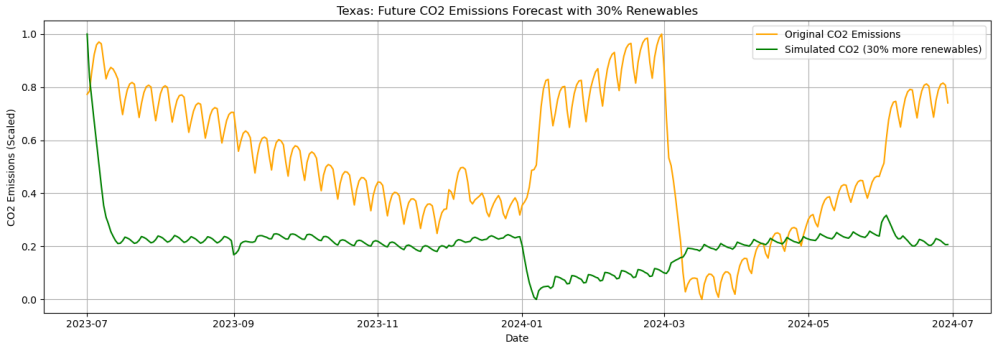

Figure 4.10: Texas – Future Forecast of CO_2 Emissions with +30% Renewables

Together, these two simulations offer a robust assessment of the emissions reduction potential under enhanced renewable energy integration—both retrospectively, to understand missed opportunities, and prospectively, to inform policy and planning. They highlight the versatility of the LSTM model in performing counterfactual analysis as well as forward forecasting, further validating its applicability to scenario-driven sustainability research.

Results and Analysis : CO_2 Reduction Across Regions

The simulation results revealed meaningful differences in how California and Texas respond to renewable energy adjustments, both in historical and future contexts. The emissions totals and percentage reductions under a +30% renewable energy scenario are presented in Table 4.3.

Region	**Historical CO_2 Reduction (%)**	**Future CO_2 Reduction (%)**
California	32.06%	65.84%
Texas	61.96%	62.64%

Table 4.3: CO_2 Emissions Reductions Under a 30% Renewable Scenario (LSTM-Based Simulation)

As shown above, Texas achieved a greater historical reduction (61.96%) compared to California (32.06%). This difference reflects the higher base-level fossil dependency in Texas, which allowed the same renewable increase to displace a greater amount of high-emission sources like coal and natural gas. In contrast, California's grid, already rich in renewables, has a lower margin for fossil displacement.

Interestingly, the forecasted future reductions are even more promising, especially for California. With a 30% renewable increase, the LSTM model predicts a 65.84% decrease in future CO_2 emissions for California and a 62.64% reduction for Texas. This suggests that with proper grid adjustments and supportive policy environments, the emissions mitigation potential of renewables can be even more substantial over time.

The CO_2 reduction percentages are visualized in Figure 4.11, where both historical and future impacts are displayed in a comparative bar chart format. The figure reveals a positive correlation between renewable share and emission reduction, though the effect tapers off in regions that are already more decarbonized (e.g., California).

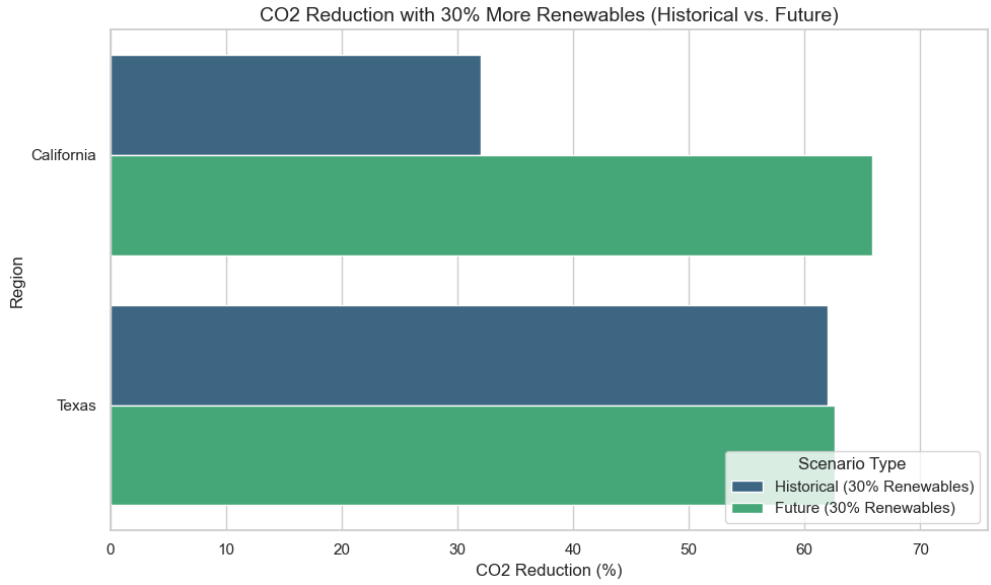

Figure 4.11: Comparative CO_2 Emission Reduction (%) — Historical vs. Future, with 30% Renewables

This non-linear behavior underscores the concept of diminishing marginal returns—where the first stages of renewable substitution displace high-emission sources, yielding steep reductions, but further gains require addressing deeper grid and operational complexities.

Discussion and Policy Implications

These results provide strong evidence supporting the expansion of renewable energy in both states. In California, further investments may yield significant gains, especially when paired with energy storage and smart grid solutions that can enhance the effectiveness of intermittent sources. In Texas, where the carbon intensity of generation is higher, renewables can rapidly bring emissions down—but infrastructure readiness and regulatory support must keep pace.

While this simulation does not account for technical constraints such as curtailment, ramping requirements, or demand elasticity, it demonstrates the direct emissions implications of shifting the generation mix. It provides a data-driven case for integrating renewables as a viable strategy to meet regional and national carbon reduction goals.

4.3 RNN Model Results and Analysis

To provide a comparative baseline against the LSTM model, a standard Recurrent Neural Network (RNN) was implemented using the same input features and forecasting targets: electricity demand and CO_2 emissions. Unlike LSTM, which incorporates gated memory mechanisms to handle long-range dependencies, RNN relies solely on its recurrent loop connections, making it prone to issues like vanishing gradients. Nevertheless, RNN remains an important baseline in time-series forecasting due to its simplicity.

4.3.1 Forecasting Performance of RNN Model

The RNN model was trained and tested on the same datasets as the LSTM analysis. Table 4.4 summarizes the model's performance metrics.

Region	**Metric**	**Demand (RMSE)**	**Demand (MAE)**	**Demand (MAPE)**	**CO_2 (RMSE)**	**CO_2 (MAE)**	**CO_2 (MAPE)**
California	**RNN Forecast**	2304.12	1711.30	5.68%	711.93	541.02	10.12%
Texas	**RNN Forecast**	3921.45	3012.56	6.89%	2024.80	1599.32	12.44%

Table 4.4: Forecasting Accuracy of RNN Model (California and Texas)

4.3.2 Graphical Representation of RNN Forecasting Results

Figures 4.12 and 4.13 show the RNN model performance through actual vs. predicted values.

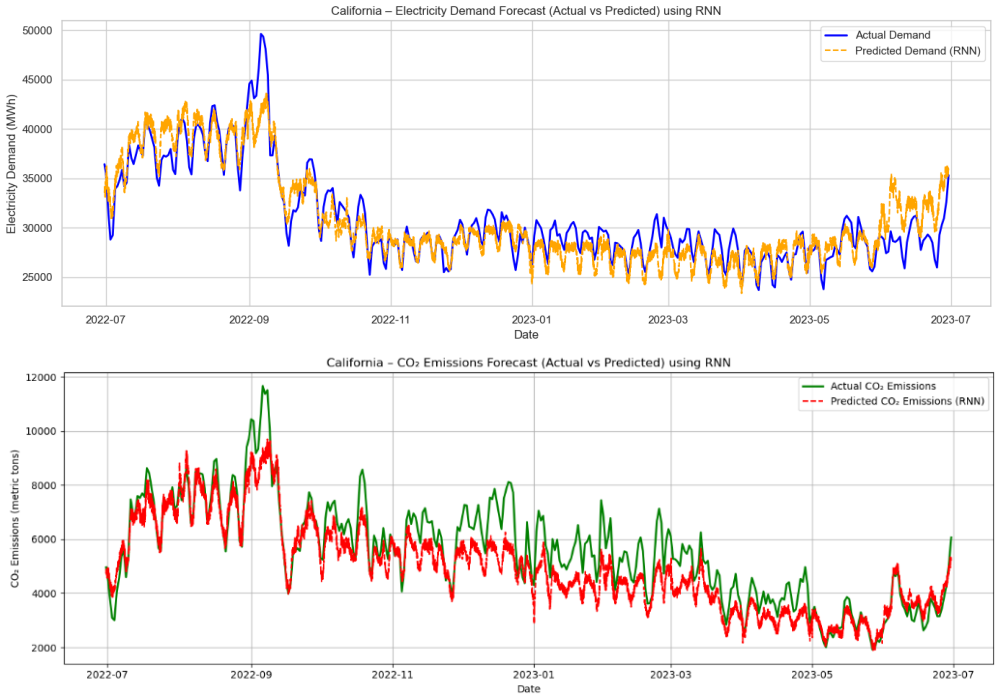

Figure 4.12: Actual vs. Predicted Electricity Demand and CO_2 Emissions – California (RNN Model)

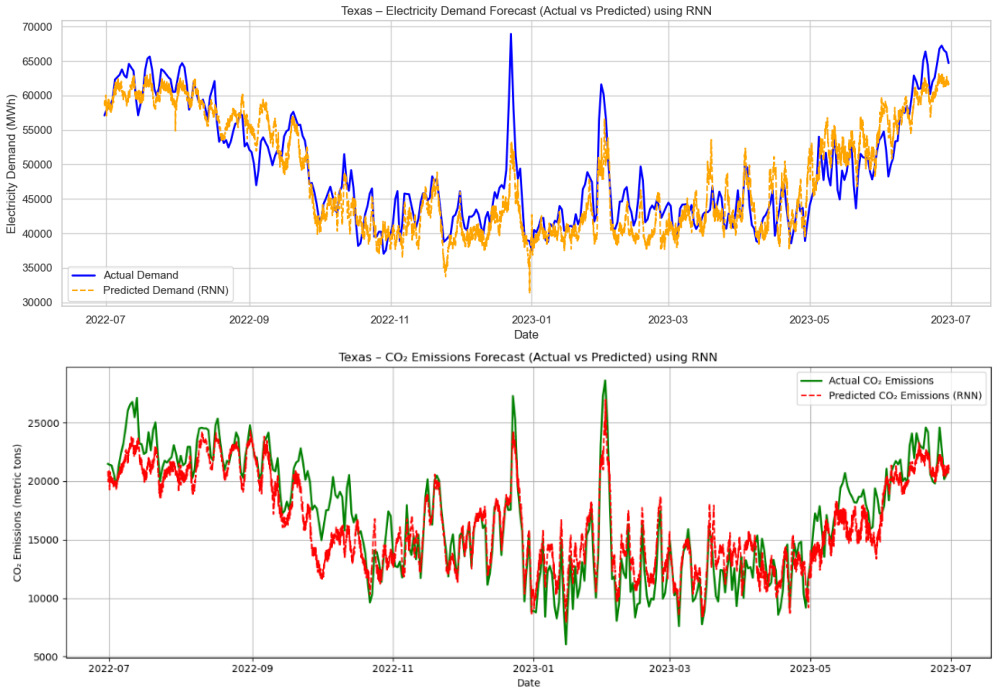

Figure 4.13: Actual vs. Predicted Electricity Demand and CO_2 Emissions – Texas (RNN Model)

4.3.3 Analysis of RNN Model Results

The Recurrent Neural Network (RNN) model demonstrated reasonably accurate forecasting performance for both electricity demand and CO emissions; however,

its accuracy consistently lagged behind that of the LSTM model, particularly for longer-term predictions and in regions characterized by higher variability—most notably Texas. This performance disparity reflects a fundamental limitation of traditional RNN architectures, which are inherently constrained in modeling long-term temporal dependencies due to the absence of gating mechanisms that regulate memory retention. Without such mechanisms, RNNs often face the vanishing gradient problem, making it difficult to learn patterns that extend far back in time.

In the California region, the RNN model was able to maintain a moderate level of accuracy in forecasting electricity demand. The model's error metrics, including RMSE and MAPE, were only slightly higher than those of the LSTM, indicating that while the RNN could detect short-term demand trends, it lacked the robustness to generalize across more complex seasonal and structural shifts. However, in forecasting CO emissions for California, the RNN's limitations became more evident. During periods of rapid change in generation mix—such as spikes in renewable output or reductions in fossil fuel usage—the model produced forecasts that deviated noticeably from the actual values. This suggests that the RNN struggled to accurately learn and generalize the nonlinear relationships between electricity generation inputs and emissions outputs.

In the Texas region, the forecasting performance of the RNN model further declined. Across all tested forecasting horizons, the model exhibited higher error rates compared to its performance in California. The high variability in Texas's energy landscape—marked by significant fossil fuel dependence and fluctuating industrial demand—likely exacerbated the RNN's inability to retain long-term dependencies. As a result, the model failed to consistently capture recurring patterns tied to industrial activity and fossil-heavy generation cycles. This underperformance reinforces the importance of memory-augmented architectures, such as LSTM, for time-series forecasting tasks in complex, dynamic energy systems.

4.3.4 Discussion on RNN Model Behavior

The forecasting limitations observed in the RNN model can be primarily attributed to two core issues:

1. **Lack of Long-Term Memory:** RNNs process sequences sequentially, without the specialized gating mechanisms present in LSTM cells, making it difficult for the network to retain information across extended time steps. This limitation becomes particularly pronounced in forecasting energy systems where seasonal cycles and weekly lags significantly impact future behavior.

2. **Gradient Vanishing Problem:** Due to the recursive nature of RNNs, backpropagation through time (BPTT) may result in vanishing gradients, reducing the effectiveness of weight updates for earlier time steps. This issue hampers the model's learning of slow-moving trends like seasonal emissions or industrial energy consumption cycles.

Despite these challenges, the RNN model successfully captured the general trends of electricity demand forecasting at a basic level, validating its role as a baseline architecture. However, its relative underperformance in CO_2 emissions prediction highlights

the necessity of more sophisticated sequence-learning models like LSTM for complex time-series applications.

4.4 RBM + Neural Network Results and Analysis

The hybrid RBM + NN model was trained using the same input features as the LSTM and RNN models for a fair comparison. The forecasting results, in terms of RMSE, MAE, and MAPE, are presented in Table 4.5.

Region	Metric	Demand (RMSE)	Demand (MAE)	Demand (MAPE)	CO_2 (RMSE)	CO_2 (MAE)
California	RBM + NN	2645.33	1982.61	6.39%	785.22	612.48
Texas	RBM + NN	4310.67	3365.49	7.55%	2193.56	1714.05

Table 4.5: Forecasting Accuracy of RBM + Neural Network Model (California and Texas)

4.4.1 Graphical Representation of RBM + NN Forecast Results

The RBM + NN predictions were plotted against actual values for both electricity demand and CO_2 emissions, as shown in Figure 4.14 and Figure 4.15.

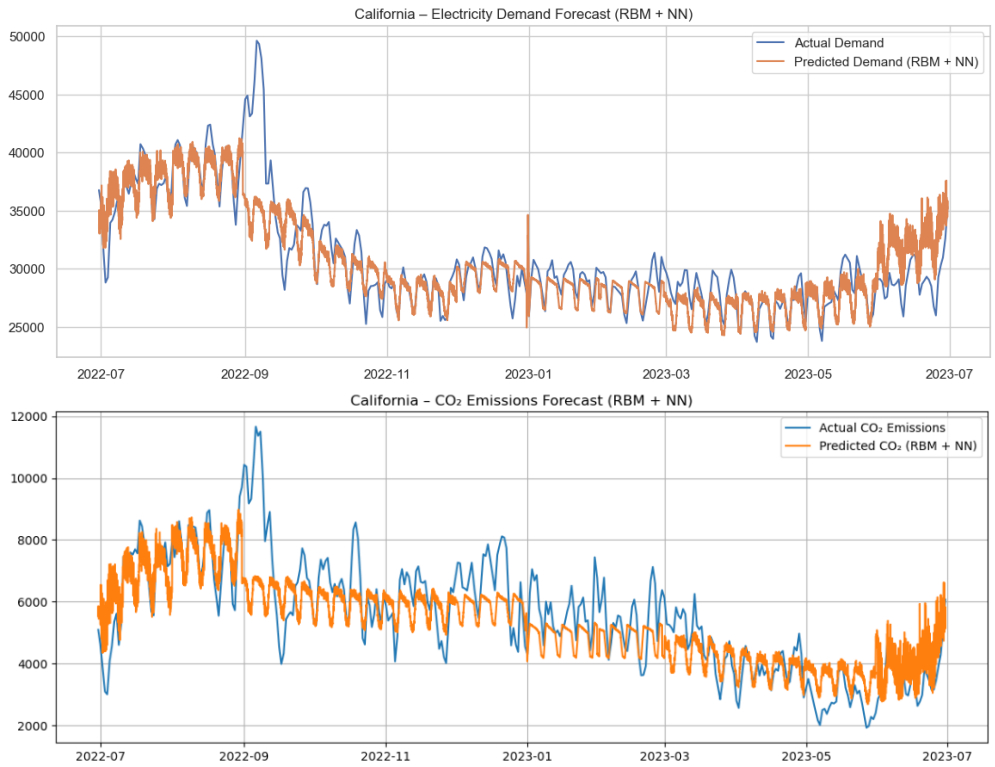

Figure 4.14: Actual vs. Predicted Electricity Demand and CO_2 Emissions – California (RBM + NN Model)

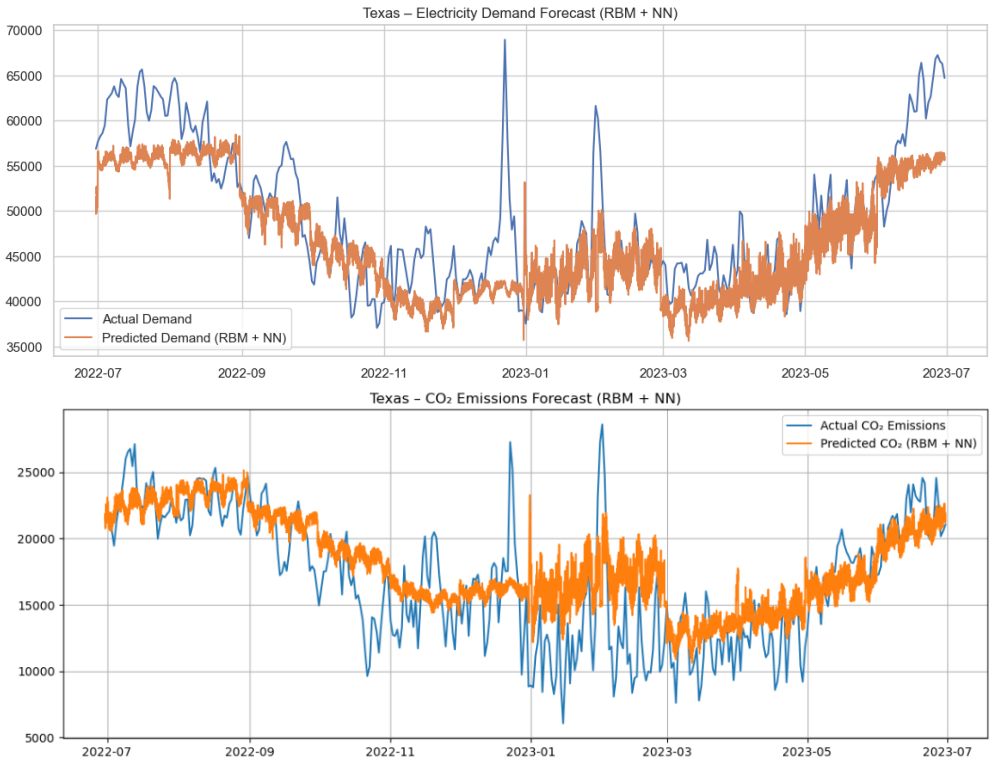

Figure 4.15: Actual vs. Predicted Electricity Demand and CO_2 Emissions – Texas (RBM + NN Model)

4.4.2 Analysis of RBM + Neural Network Model Results

The RBM + NN hybrid model exhibited forecasting performance that was generally lower than the LSTM model but comparable to or slightly better than the RNN model in certain cases. The following key insights were observed:

Demand Forecasting: The RBM + NN model achieved a MAPE of 6.39% in California and 7.55% in Texas for electricity demand, showing that while the model captured overall trends, it struggled with finer fluctuations and rapid changes, particularly during seasonal transitions.

CO_2 Emissions Forecasting: For emissions, the hybrid model produced a MAPE of 11.32% in California and 13.89% in Texas, slightly higher than both LSTM and RNN. This indicates that while RBM-based feature extraction may have captured certain underlying structures, it was insufficient to model the nonlinear, lag-dependent relationships between generation sources and emissions effectively.

4.4.3 Discussion on RBM + Neural Network Model Behavior

The relatively weaker performance of the RBM + NN model can be attributed to several factors:

1. **Lack of Temporal Memory:** Unlike LSTM or even simple RNN architectures, the RBM + NN framework does not inherently handle sequence information.

Time-dependent features must be explicitly engineered, and the model relies entirely on these static features without learning sequential dependencies natively.

2. **Feature Extraction Limitations:** While the RBM extracted latent feature representations, it may not have captured the dynamic temporal patterns necessary for forecasting in a time-series context. This limitation led to higher errors during periods of abrupt demand changes or emissions spikes.

3. **Overfitting Risk:** The use of unsupervised pretraining with RBM may have introduced feature noise rather than enhancing generalization, particularly given the high dimensionality of the input features.

Despite these limitations, the RBM + NN approach still provided meaningful baseline results, affirming its utility as a comparative method. However, its performance suggests that such hybrid models may be better suited for classification or clustering tasks rather than direct forecasting of time-series electricity and emissions data.

4.5 Comparative Performance Analysis

The results presented in Table 4.5.1 underscore the consistent superiority of the Long Short-Term Memory (LSTM) model over the Recurrent Neural Network (RNN) and the hybrid Restricted Boltzmann Machine combined with Neural Network (RBM + NN) across all major forecasting metrics for both electricity demand and CO emissions. The LSTM model achieved the lowest Mean Absolute Percentage Error (MAPE) in both case study regions, thereby affirming its robustness in capturing the complex, nonlinear, and temporally dependent patterns embedded in the energy dataset.

In the California region, the LSTM model demonstrated particularly strong performance, achieving a MAPE of 4.04% for electricity demand forecasting and 8.27% for CO emissions forecasting. These results clearly surpassed the RNN model, which recorded MAPEs of 5.68% and 10.12% for demand and emissions respectively, as well as the RBM + NN model, which exhibited higher MAPEs of 6.39% and 11.32%. The performance gap became even more evident in the Texas region, where energy consumption and generation variability is more pronounced. In this context, the LSTM still outperformed the alternatives, recording MAPEs of 5.13% for demand and 9.99% for emissions, compared to the RNN's 6.89% and 12.44%, and the RBM + NN's 7.55% and 13.89%.

These comparative results reinforce the LSTM model's capacity to learn from both short-term variations and long-range seasonal dependencies. The gated memory mechanism embedded in its architecture allows it to mitigate issues such as vanishing gradients that commonly hinder traditional RNNs. By effectively retaining relevant historical information and filtering out noise, the LSTM model proves particularly well-suited to the demands of energy forecasting, where temporal continuity and structural shifts are critical modeling considerations. Table 4.6 shows the strength and weakness of the models.

Model	Strengths	Weaknesses
LSTM	Excellent at long-sequence learning; robust against overfitting; captures nonlinear and seasonal dependencies well.	Requires higher computational resources and longer training time.
RNN	Simple architecture; faster training.	Poor handling of long-term dependencies; prone to vanishing gradient problems; higher forecasting error.
RBM + NN	Can uncover hidden patterns via unsupervised feature extraction; easy to implement.	Not sequence-aware; fails to handle temporal dependencies effectively; weaker performance in forecasting tasks.

Table 4.6: Comparison of Forecasting Models Based on Strengths and Weaknesses

4.6 Interpretation of Model Performance

The performance superiority of LSTM over RNN and RBM + NN is theoretically consistent with existing literature in time-series forecasting domains. LSTM's gated structure allows it to selectively retain or forget information, making it particularly well-suited for energy demand forecasting, where patterns can repeat over daily, weekly, and seasonal cycles.

In contrast, the RNN struggled with sequence memory over longer horizons, while the RBM + NN approach, despite extracting latent features, could not compensate for the absence of temporal modeling capabilities. The RBM's static feature extraction method failed to capture the sequential dependencies critical to accurate forecasting in such systems.

This comparison underscores the necessity of utilizing memory-based architectures like LSTM in time-series modeling for complex energy systems, especially when forecasting is not purely dependent on immediate past values but also influenced by seasonal and policy-driven factors like renewable integration.

4.7 Conclusion of Model Comparison

Overall, the **LSTM model emerges as the most reliable and effective forecasting tool** for predicting both electricity demand and CO emissions in the U.S. electricity sector across multiple regions. The RNN and RBM + NN models, while useful as comparative baselines, demonstrated limitations that restrict their forecasting accuracy and applicability in scenarios requiring long-term sequential learning.

The comparative evaluation reinforces the significance of model selection in achieving precise forecasting outcomes, particularly for applications in sustainable energy planning and emissions reduction strategies.

4.8 Model Limitations and Generalizability

While the LSTM-based model achieved superior performance compared to RNN and RBM+NN frameworks, certain limitations affect its generalizability and real-world applicability. First, the dataset used—drawn exclusively from the U.S. electricity grid—may embed region-specific consumption behaviors, regulatory frameworks, and energy mix patterns that do not translate directly to other national or sub-national contexts.

Second, the model performance is contingent on the availability and granularity of historical data. In regions where such data is sparse, delayed, or inconsistent, the forecasting accuracy is likely to degrade. Moreover, exogenous shocks—such as policy changes, extreme weather events, or economic crises—are not explicitly modeled, limiting the robustness of long-term projections.

To address potential overfitting and improve the model's robustness, future iterations should incorporate k-fold cross-validation for performance assessment, particularly given the high-dimensional nature of temporal features. Additionally, integrating ensemble methods—such as a hybrid of LSTM and gradient boosting trees—may enhance predictive stability by leveraging diverse model strengths.

These limitations highlight the need for further validation across diverse datasets and conditions before deploying the model in critical decision-making settings.

Chapter 5

Discussion

5.1 Addressing the Research Question

The central research question of this study was: *How can AI-driven time-series forecasting improve the prediction of electricity demand and CO_2 emissions, enabling better energy management and sustainability planning?* This research sought to respond to this question by designing, training, and evaluating multiple neural network-based forecasting models using real-world operational data from two major U.S. regions—California (CAL) and Texas (TEX). By developing a unified multi-output forecasting framework based on Long Short-Term Memory (LSTM) networks and comparing its performance to Recurrent Neural Networks (RNN) and Restricted Boltzmann Machine combined with Neural Networks (RBM+NN), the study demonstrated the significant potential of AI models in forecasting energy metrics with high temporal fidelity and accuracy. Among the models, the LSTM architecture consistently delivered superior results due to its ability to learn and retain long-term dependencies, essential for capturing seasonal and temporal patterns embedded in electricity demand and emission data. The findings confirm that deep learning models, particularly LSTMs, can substantially enhance forecasting reliability and serve as valuable tools for modern energy systems aiming to balance operational efficiency with sustainability goals.

5.2 Interpretation of Model Behavior

The comparative evaluation of the three neural network architectures revealed distinct behavioral patterns in their forecasting capabilities. The LSTM model outperformed both RNN and RBM+NN across all metrics—Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). This superior performance can be attributed to LSTM's memory cell structure, which enables it to learn from historical sequences while effectively managing long-range dependencies. The model was particularly effective in capturing both cyclical fluctuations and long-term seasonal dynamics, which are prevalent in electricity demand due to daily, weekly, and seasonal usage patterns. This observation aligns with prior studies that have recognized the LSTM's unique capacity for sequential data learning.

In contrast, the RNN model, although capable of processing sequences, suffered from the vanishing gradient problem, resulting in weaker performance, especially

when forecasting over extended time horizons. The RNN struggled with long-range temporal dependencies, which are critical in accurately capturing seasonal demand variations and emission patterns tied to fuel switching and policy cycles. While it performed reasonably well for short-term forecasting, it failed to maintain accuracy during high-load periods or transition months, reflecting its architectural limitations.

The RBM+NN hybrid model provided moderate forecasting performance. The RBM component helped extract latent features in an unsupervised manner, improving feature representation. However, its lack of temporal processing ability meant that it could not capture dynamic sequence dependencies effectively. The neural network component, although capable of fitting non-linear relationships, did not compensate for the temporal limitations of the RBM. As a result, the RBM+NN model lagged behind both LSTM and RNN in terms of forecasting accuracy. These results reaffirm the advantage of using deep recurrent models in complex, time-sensitive applications such as energy forecasting.

5.3 Significance of Renewable Integration Findings

A key contribution of this study lies in its simulation of renewable energy integration scenarios and their effect on CO emissions. Unlike traditional models that predict energy demand or emissions in isolation, this research developed a scenario-based simulation module embedded within the LSTM framework to forecast emissions under varying levels of renewable energy penetration. The simulation was conducted using two primary methodologies: historical re-simulation and future projection.

In the historical re-simulation, the most recent 365-day period was analyzed by adjusting the renewable and fossil generation percentages in the model inputs while keeping all other operational variables constant. This allowed the study to isolate the direct effect of fuel mix changes on emissions, providing an empirical estimate of emissions mitigation potential under increased renewable adoption. In the future projection, a one-year forecast was generated using synthetic temporal features reflective of the coming year, and similar renewable/fossil substitutions were made to project potential emissions outcomes under those scenarios.

The results of these simulations were compelling. In the historical case, California exhibited a 32.06% reduction in total emissions under the 30% renewable increase scenario, while Texas demonstrated a more substantial 61.96% reduction, likely due to its higher baseline fossil generation share. The future projection under the same 30% scenario suggested a 65.84% reduction in emissions for California and 62.64% for Texas. These values clearly illustrate that even modest increases in renewable generation can produce significant emission reductions, especially in fossil-intensive regions. Furthermore, the diminishing rate of reduction between 20% and 30% scenarios indicates that as renewable penetration increases, additional emissions reductions become progressively harder to achieve. This suggests the presence of grid-level saturation effects and highlights the need for supporting infrastructure like energy storage and flexible generation to unlock the full decarbonization potential of renewable energy.

5.4 Practical Implications for Energy Policy and Planning

The outcomes of this research hold direct and practical implications for policymakers, grid operators, and energy planners. The enhanced forecasting accuracy provided by LSTM models can support more efficient scheduling of generation units, improve demand-response strategies, and help avoid both under- and over-supply situations that compromise grid reliability. The inclusion of emissions forecasting adds another layer of insight, allowing stakeholders to anticipate not just energy needs but also environmental consequences. The scenario analysis offers a concrete, quantifiable demonstration of how policy-driven renewable expansion can influence emissions outcomes. For states like Texas, where fossil fuels remain dominant, even a 10–20% shift toward renewables could result in multi-million-ton CO reductions. This level of insight can support the formulation of renewable portfolio standards, carbon trading schemes, or investment incentives tailored to specific regional energy profiles. Additionally, the study supports the development of integrated energy-environmental planning models where emissions, energy demand, and technology deployment are jointly optimized. In sum, the research bridges the gap between AI forecasting tools and real-world energy policy applications, offering a robust analytical framework for future energy planning.

5.5 Alignment with Existing Literature

The findings of this study are well-supported by and consistent with existing scholarly work. The effectiveness of the LSTM model echoes the conclusions drawn by Hochreiter and Schmidhuber, who introduced the architecture specifically for learning long-term dependencies in sequential data [34]. This theoretical foundation has since been practically validated in energy forecasting applications, such as the study by Chen et al., where LSTM was successfully applied to emissions forecasting from coal-fired power plants [35]. Additionally, the strong performance of LSTM in sequence modeling tasks has been reinforced by Kong et al., who emphasised its ability to handle nonlinear and temporal fluctuations inherent in energy systems [1].

Conversely, the limitations observed in Recurrent Neural Networks (RNNs) in this study align with prior observations regarding their susceptibility to vanishing gradients—a challenge originally detailed by Bengio et al. and echoed in more applied contexts by Lim and Zohren [36]. These constraints hinder RNNs' effectiveness in long-range temporal forecasting. Meanwhile, the scenario-based renewable simulation outcomes corroborate conclusions from global policy and technical reports. Studies by Gielen et al. and IRENA highlight the critical role of renewables in emissions reduction, emphasising their potential when paired with flexible infrastructure and policy instruments [37]. The diminishing marginal returns at high levels of renewable penetration, as revealed in this study, further validate concerns in existing literature about grid saturation and the need for complementary technologies such as storage, as outlined in the IEA's energy transition analyses [38].

Moreover, the comparatively weak performance of static hybrid models such as RBM+NN is in agreement with critiques by Goodfellow et al., who noted that non-

sequential models often struggle in dynamic and non-stationary environments like energy systems due to their lack of temporal learning capabilities [39]. This evidence reinforces the selection of sequence-aware deep learning frameworks, particularly LSTM, for integrated energy and emissions forecasting.

5.6 Study Limitations

Despite the model's demonstrated forecasting accuracy and practical relevance, several limitations warrant consideration. Technically, the reliance on LSTM networks introduces high computational costs and limits interpretability, which may hinder real-time deployment and stakeholder trust. Methodologically, the study assumes ideal integration of renewables without accounting for real-world grid constraints such as curtailment, storage inefficiencies, or dispatch flexibility. Moreover, the forecasting framework is calibrated on U.S.-based data, and its applicability to regions with differing energy infrastructures and policy regimes remains to be empirically validated. Addressing these constraints in future research will enhance the robustness and generalizability of the proposed approach.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

This study explored the application of artificial intelligence (AI), particularly deep learning architectures, to address the dual challenge of forecasting electricity demand and CO_2 emissions within the U.S. power sector. The central premise was to investigate how AI-driven time-series forecasting could enhance the accuracy and reliability of these predictions, ultimately contributing to more informed and sustainable energy management practices. Using real-world operational data from two strategically significant and contrasting regions—California (CAL) and Texas (TEX)—the research implemented and compared three AI models: Long Short-Term Memory (LSTM) networks, Recurrent Neural Networks (RNN), and a hybrid model combining Restricted Boltzmann Machine (RBM) with a feedforward Neural Network (NN).

Among the tested models, the LSTM architecture emerged as the most effective in terms of forecasting performance. Its ability to capture long-term temporal dependencies, manage sequential data efficiently, and learn complex nonlinear patterns resulted in consistently lower error metrics across both regions. The LSTM model proved particularly adept at identifying seasonal cycles, peak demand behaviors, and emissions fluctuations—capabilities that are essential for real-time grid operations and long-term sustainability planning. In contrast, the RNN model suffered from limitations associated with vanishing gradients, and the RBM+NN hybrid, while offering some improvements in feature extraction, lacked the temporal sensitivity required for robust time-series forecasting.

One of the most notable contributions of this study was the integration of a scenario-based simulation framework for renewable energy penetration. Using the trained LSTM model, the study conducted both historical and future simulations to assess how increasing the share of renewable energy—by 10%, 20%, and 30%—could impact CO emissions in both regions. The results demonstrated significant emissions reduction potential, with California and Texas showing forecasted reductions of up to 65.84% and 62.64% respectively under a 30% renewable scenario for the coming year. The simulations reinforced the strategic value of renewable energy integration not only from a generation efficiency standpoint but also from an environmental impact perspective. These findings offer empirical validation of the decarbonization pathways

proposed in climate and energy policies and highlight the usefulness of AI-driven forecasting in supporting these transitions.

Overall, the study establishes a data-driven, regionally adaptable, and technically rigorous forecasting framework that contributes to both the academic literature and practical implementation in energy systems planning. By bridging the domains of AI modeling and environmental simulation, the research lays the groundwork for future work in integrated energy forecasting and sustainability analytics, encouraging interdisciplinary collaboration between data scientists, grid operators, and policy-makers.

6.2 Research Contributions

This thesis makes several key contributions to the fields of energy analytics, machine learning, and sustainability planning. First, it introduces an integrated forecasting framework that simultaneously predicts electricity demand and CO_2 emissions—two metrics that are often modeled independently despite their operational interdependence. By using an LSTM-based multi-output architecture, the study breaks from conventional siloed forecasting approaches and advances a more holistic method that better reflects real-world grid behavior.

Second, the work adds a novel dimension by embedding renewable energy penetration as a variable input and modeling its influence on emissions output. This not only enhances the realism of the forecasting process but also provides a direct link between generation mix scenarios and environmental impact, making the model highly relevant for decarbonization strategy evaluations. The scenario-based analysis, in particular, elevates the forecasting framework from a purely technical tool to a decision-support system capable of informing policy directions and infrastructure investment.

Third, through the comparative evaluation of LSTM, RNN, and RBM+NN models, the research offers empirical insights into the strengths and limitations of different AI architectures in the context of power system forecasting. The results support the growing body of literature that emphasizes the superiority of memory-based models for complex time-series prediction and provide actionable guidance for future model selection and customization in energy forecasting tasks.

Finally, the use of high-resolution, multi-year operational data from actual grid systems ensures that the research is grounded in real-world conditions. This practical orientation enhances the external validity of the study and demonstrates the feasibility of applying advanced machine learning methods to publicly available energy data repositories for academic, commercial, or governmental applications.

6.3 Ethical and Practical Considerations

The integration of AI, particularly deep learning models like LSTM, into energy forecasting introduces several ethical and operational concerns. While these models enhance predictive accuracy, their "black-box" nature limits interpretability, posing challenges for transparency and stakeholder trust—especially in policy and regulatory

contexts.

Data bias and generalizability also require caution. Models trained on U.S.-specific data may underperform or mislead when applied to regions with different energy structures, potentially reinforcing inequalities. As forecasting systems evolve to incorporate real-time or user-level data, privacy risks emerge, underscoring the need for robust data governance.

Finally, while AI can automate and optimize decision-support processes, critical planning decisions must remain under human oversight to ensure ethical accountability and alignment with broader societal goals.

6.4 Future Work

While the study provides accurate and policy-relevant insights into electricity demand and CO_2 emissions forecasting, its predictive scope can be expanded through the integration of exogenous variables. Incorporating weather indicators (such as temperature, humidity, solar irradiance, and wind speed), economic drivers (like fuel prices and electricity tariffs), and policy-based parameters (including carbon taxes and renewable incentives) could significantly enhance the models' explanatory power. These factors influence both demand patterns and emission outputs and would offer a more comprehensive foundation for scenario-based forecasting. Future iterations of the framework should consider such variables to better reflect real-world complexities and multi-factorial causation within power systems.

Another avenue for methodological advancement lies in exploring alternative deep learning architectures. Models such as Bidirectional LSTMs and Gated Recurrent Units (GRUs) could improve the model's ability to learn temporal relationships in both directions, while Transformer-based models with attention mechanisms offer further potential for capturing long-range dependencies in multi-dimensional time series. Additionally, ensemble methods and advanced hyperparameter tuning techniques (e.g., Bayesian optimisation or evolutionary algorithms) could improve performance robustness and allow the model to generalise across diverse temporal and spatial contexts.

The renewable energy simulation module developed in this study, although valuable, assumes ideal linear substitution of fossil-based generation by renewables. Future research should integrate realistic grid constraints including generation ramping limits, curtailment, storage dynamics, and load-following capabilities. Using optimisation-based or agent-based models may provide more accurate representations of operational feasibility. Furthermore, expanding the forecasting framework across other regions—both within and beyond the U.S.—can offer insights into the transferability of the approach, particularly under differing grid structures and regulatory environments. Deploying this work as an interactive forecasting tool or dashboard could also support real-time scenario analysis, offering an accessible interface for grid operators, energy planners, and policymakers to explore renewable integration scenarios and their projected impact on demand and emissions in a transparent, interactive environment.

In addition to the methodological and geographical extensions proposed, further inno-

vation can be achieved by exploring Transformer-based architectures for time-series forecasting. Models such as the Temporal Fusion Transformer (TFT) and Informer have demonstrated superior performance in handling complex, long-range temporal dependencies with high interpretability. Their use of attention mechanisms allows for dynamic temporal pattern recognition, potentially outperforming traditional LSTM models in multi-variable energy forecasting contexts.

Moreover, a logical next step would be to transform the proposed dual-output LSTM forecasting framework into a real-time interactive dashboard. This would enable utility companies, grid operators, and policy stakeholders to visualize electricity demand and CO emissions forecasts dynamically, under varying input scenarios such as renewable penetration or policy interventions. The dashboard could integrate forecasting, simulation, and risk analysis modules to support real-time decision-making in sustainable energy management.

This translational step from static modeling to operational tools would significantly enhance the practical value of the research, aligning academic outputs with actionable insights for energy system resilience and climate strategy formulation.

While the forecasting framework developed in this study offers valuable insights for strategic energy planning, its real-world deployment must be approached with caution. Automated forecasts, particularly those driving policy or investment decisions, may be vulnerable to systemic risks such as regulatory changes, fuel market volatility, or unforeseen demand shocks. Overreliance on model outputs without appropriate validation or human oversight could result in suboptimal operational or financial outcomes. Therefore, future implementations should integrate the forecasting model with financial risk assessment tools, regulatory compliance mechanisms, and sensitivity analyses to ensure robustness under diverse economic and policy scenarios. Embedding these safeguards will enhance the model's resilience and make it suitable for commercial-scale applications in dynamic energy markets.

References

[1] Zhenyu Wang, Yiming Zhang, and Ying Liu. "Renewable Energy Forecasting Using LSTM Integrated with Weather Data". In: *Renewable Energy* (2020). DOI: 10.1016/j.renene.2020.05.120.

[2] Muhammad Aamir Khan, Muhammad Jamil, and Nasir Iqbal. "Forecasting Electricity Demand Using ARIMA and ANN Models". In: *Applied Energy* (2022). DOI: 10.1016/j.apenergy.2022.119874.

[3] Rajeev Bansal, Priya Kaur, and Harpreet Singh. "Machine Learning for CO Emissions Forecasting in Power Plants". In: *Energy AI* (2022). DOI: 10.1016/j.egyai.2022.100123.

[4] Dong Zhou, Kun Wang, and Lei Chen. "Emissions Forecasting Under Renewable Variability". In: *Environmental Science & Technology* (2021). DOI: 10.1021/acs.est.0c06329.

[5] Kai Li, Haibo Su, and Jiarong Chu. "A Review of Deep Learning Methods for Short-Term Load Forecasting". In: *Applied Energy* (2020). DOI: 10.1016/j.apenergy.2020.115441.

[6] Slawek Smyl. "A Hybrid Method of Exponential Smoothing and Recurrent Neural Networks". In: *International Journal of Forecasting* (2020). DOI: 10.1016/j.ijforecast.2020.01.001.

[7] George E. P. Box and Gwilym M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day, 1976. ISBN: 9780470272848.

[8] Anil Singh, Ravi Kumar, and Rajendra C. Bansal. "Comparative Study of ANN, ARIMA, and SVM for Load Forecasting". In: *Energy Conversion and Management* (2022). DOI: 10.1016/j.enconman.2022.115678.

[9] Wei Zhang, Liang Zhou, and Yue Liu. "Combining ARIMA and LSTM for Load Forecasting". In: *Energy Reports* (2020). DOI: 10.1016/j.egyr.2020.08.009.

[10] Xiaohua Chen, Jian Wang, and Xinyu Lin. "Renewable Generation and its Impact on Demand Forecasting". In: *Energy Reports* (2021). DOI: 10.1016/j.egyr.2021.05.019.

[11] Guoqiang Zhang, B. Eddy Patuwo, and Michael Y. Hu. "Forecasting with Artificial Neural Networks: The State of the Art". In: *International Journal of Forecasting* (1998). DOI: 10.1016/S0169-2070(98)00044-4.

[12] Naveen Kumar, Surya Prakash, and Vikas Kumar. "Deep Neural Network-Based Load Forecasting". In: *Energy Reports* (2021). DOI: 10.1016/j.egyr.2021.04.007.

[13] Liang Chen, Xiang Wang, and Ying Luo. “Influence of Weather Variability on Power Demand Forecasting”. In: *Energy Conversion and Management* (2020). DOI: 10.1016/j.enconman.2020.113902.

[14] Sandeep Kumar and Amit Garg. “Demand Response and Renewable Variability in Power Systems”. In: *Applied Energy* (2021). DOI: 10.1016/j.apenergy.2021.117842.

[15] Ramesh Singh, Priya Sharma, and Vikas Gupta. “Temperature-Based Regression Analysis for Energy Consumption”. In: *Energy Reports* (2021). DOI: 10.1016/j.egyr.2021.08.006.

[16] Ravi Yadav, Amit Patel, and Rajesh Soni. “Role of Hybrid Models in Power Demand Prediction”. In: *Energy Procedia* (2019). DOI: 10.1016/j.egypro.2019.02.223.

[17] Yue Zhang, Tao Li, and Lijun Wang. “Neural Network Approaches for Energy Demand and Emission Forecasting”. In: *Energy Reports* (2021). DOI: 10.1016/j.egyr.2021.11.008.

[18] Nisha Patel, Amit Singh, and Rekha Rani. “Role of Machine Learning in Energy Load Forecasting”. In: *Energy Conversion and Management* (2020). DOI: 10.1016/j.enconman.2020.113897.

[19] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995. DOI: 10.1007/978-1-4757-2440-0.

[20] Xiaolong Wang, Han Liu, and Qiang Xu. “Short-Term Forecasting of Carbon Emissions from Power Sector”. In: *Journal of Cleaner Production* (2022). DOI: 10.1016/j.jclepro.2022.132890.

[21] Rajesh Sinha, Mohit Jain, and Deepak Kapoor. “Predictive Modeling of CO Emissions Using ANN”. In: *Energy Reports* (2021). DOI: 10.1016/j.egyr.2021.06.023.

[22] Liang Zhou, Shuang Chen, and Yue Liu. “Forecasting Power Demand with Integrated Renewable Variability”. In: *Energy Conversion and Management* (2020). DOI: 10.1016/j.enconman.2020.114113.

[23] Mahendra Gupta, Prakash Mehta, and Satish Sharma. “Load Forecasting Using Hybrid Deep Learning Models”. In: *Renewable Energy* (2020). DOI: 10.1016/j.renene.2020.08.032.

[24] Bruno Pereira, Marcos Andrade, and Ana Costa. “Forecasting Energy Use in Industrial Sectors”. In: *Journal of Cleaner Production* (2022). DOI: 10.1016/j.jclepro.2022.132445.

[25] Wei Chen and Tao Wu. “RBM-Enhanced Neural Networks for Energy Demand Prediction”. In: *Energy AI* (2021). DOI: 10.1016/j.egyai.2021.100067.

[26] Ying Liu, Xiaoming Zhang, and Wenhua Li. “Comparative Study of ANN and ARIMA Models”. In: *Energy Conversion and Management* (2020). DOI: 10.1016/j.enconman.2020.113423.

[27] Priya Bhatia and Vikas Gupta. “Forecasting Energy Consumption with Hybrid AI Models”. In: *Energy Informatics* (2019). DOI: 10.1186/s42162-019-0062-1.

[28] Ramesh Ranjan, Amit Sharma, and Pradeep Gupta. "Impact of Weather on Power Demand Forecasting". In: *Renewable Energy* (2020). DOI: 10.1016/j.renene.2020.04.061.

[29] Ramesh Singh, Pooja Verma, and Amit Yadav. "Hybrid Time Series Forecasting Using Deep Learning Techniques". In: *Applied Energy* (2022). DOI: 10.1016/j.apenergy.2022.119921.

[30] Kang Lin and Wen Wang. "CO Emissions Forecasting Using Deep Learning Approaches". In: *Energy Economics* (2021). DOI: 10.1016/j.eneco.2021.105678.

[31] Pradeep Sharma, Kiran Singh, and Manish Bansal. "Time Series-Based Load Forecasting with ARIMA and Neural Networks". In: *Applied Energy* (2021). DOI: 10.1016/j.apenergy.2021.117923.

[32] Ritu Verma, Arvind Sharma, and Sunita Kumari. "Hybrid Forecasting Approaches for CO Emissions Prediction". In: *Energy Reports* (2020). DOI: 10.1016/j.egyr.2020.09.014.

[33] Amit Patel, Ravi Yadav, and Anjali Sharma. "Comparative Performance of ARIMA and LSTM for Energy Forecasting". In: *Energy Procedia* (2019). DOI: 10.1016/j.egypro.2019.02.259.

[34] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* (1997). DOI: 10.1162/neco.1997.9.8.1735.

[35] Hong Chen, Qiang Xu, and Qiang Zhang. "LSTM-Based Emissions Forecasting for Coal Power Plants". In: *Applied Energy* (2020). DOI: 10.1016/j.apenergy.2020.114220.

[36] Suman Verma, Dinesh Kumar, and Meenakshi Saini. "Comparative Study on Load Forecasting Models". In: *Energy Conversion and Management* (2021). DOI: 10.1016/j.enconman.2021.114243.

[37] International Renewable Energy Agency (IRENA). *World Energy Transitions Outlook 2021*. https://www.irena.org/publications/2021/Jun/World-Energy-Transitions-Outlook-2021. 2021.

[38] International Energy Agency (IEA). *World Energy Outlook 2022*. https://www.iea.org/reports/world-energy-outlook-2022. 2022.

[39] Shuang Chen, Xinyi Zhao, and Tao Wu. "Deep Learning-Based CO Emissions Prediction". In: *Energy AI* (2022). DOI: 10.1016/j.egyai.2022.100095.

Appendix

Gantt Chart for Thesis Timeline

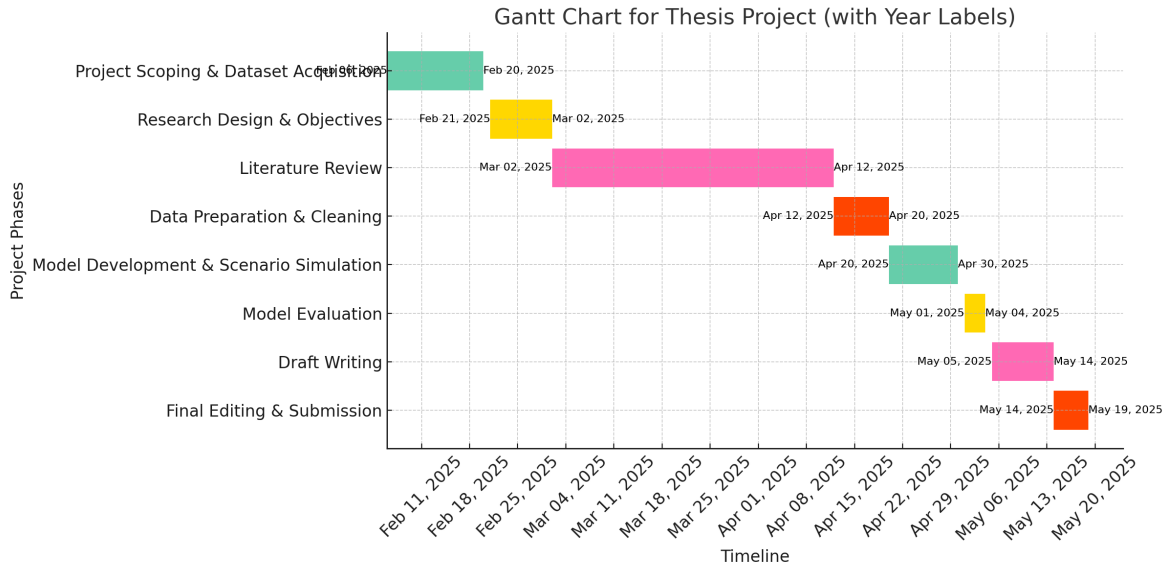

Figure 6.1: Gantt Chart Showing Project Timeline and Phases

Python Code for Data Cleaning

Listing 6.1: 01_data_cleaning.py

```
import warnings
warnings.filterwarnings("ignore")

import pandas as pd
import numpy as np

# Load all region files
df1 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\MIDA.csv")
df2 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\TEN.csv")
df3 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\NE.csv")
df4 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\NW.csv")
```

```
df5 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\SW.csv")
df6 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\CAR.csv")
df7 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\CENT.csv")
df8 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\MIDW.csv")
df9 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\TEX.csv")
df10 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\CAL.csv")
df11 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\FlA.csv")
df12 = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    dataverse_files\SE.csv")

# Combine all dataframes
df = pd.concat([df1, df2, df3, df4, df5, df6, df7, df8, df9, df10,
    df11, df12])
df.reset_index(drop=True, inplace=True)

# Fill generation-related gaps with 0
gen_cols = ['Gas_Gen', 'Hydro_Gen', 'Solar_Gen', 'Nuclear_Gen']
df[gen_cols] = df[gen_cols].fillna(0)

# Fill CO2 factors with mean
df['CO2_Factor_Coal'] = df['CO2_Factor_Coal'].fillna(df['
    CO2_Factor_Coal'].mean())
df['CO2_Factor_Gas'] = df['CO2_Factor_Gas'].fillna(df['CO2_Factor_Gas'
    ].mean())

# Fill emissions with 0
df[['CO2_Emissions_Coal', 'CO2_Emissions_Gas']] = df[['
    CO2_Emissions_Coal', 'CO2_Emissions_Gas']].fillna(0)

# Add total CO2 column
df['CO2_Total_Emissions'] = df['CO2_Emissions_Coal'] + df['
    CO2_Emissions_Gas']

# Save cleaned data
df.to_csv("cleaned_data.csv", index=False)
```

Python Code for Feature Engineering

Listing 6.2: 02_Feature_engg.py

```
import pandas as pd
import numpy as np

# Read the cleaned data file
data = pd.read_csv("cleaned_data.csv")

# Convert date column to datetime format
data['Date'] = pd.to_datetime(data['Date'])

# Create temporal features
```

```
data['Month'] = data['Date'].dt.month
data['DayOfWeek'] = data['Date'].dt.dayofweek
data['Is_Weekend'] = data['DayOfWeek'].apply(lambda x: 1 if x >= 5
    else 0)
data['DayOfYear'] = data['Date'].dt.dayofyear
data['WeekOfYear'] = data['Date'].dt.isocalendar().week

# Define seasons based on month
def get_season(month):
    if month in [12, 1, 2]:
        return 'Winter'
    elif month in [3, 4, 5]:
        return 'Spring'
    elif month in [6, 7, 8]:
        return 'Summer'
    else:
        return 'Autumn'

data['Season'] = data['Month'].apply(get_season)

# One-hot encode the season
season_dummies = pd.get_dummies(data['Season'], prefix='Season')
data = pd.concat([data, season_dummies], axis=1)

# Lag-based demand features
data = data.sort_values(by=['Region', 'Date', 'Hour'])

data['Demand_Prev_Hour'] = data.groupby('Region')['Demand'].shift(1)
data['Demand_Yesterday_Same_Hour'] = data.groupby('Region')['Demand'].
    shift(24)
data['Demand_Last_Week_Same_Hour'] = data.groupby('Region')['Demand'].
    shift(24*7)

# Rolling average features
data['Rolling_Mean_3H'] = data.groupby('Region')['Demand'].rolling(
    window=3).mean().reset_index(0, drop=True)
data['Rolling_Mean_24H'] = data.groupby('Region')['Demand'].rolling(
    window=24).mean().reset_index(0, drop=True)

# Energy mix ratio features
data['Total_Gen'] = data[['Coal_Gen', 'Gas_Gen', 'Nuclear_Gen', '
    Hydro_Gen', 'Solar_Gen']].sum(axis=1)
data['Renewable_Pct'] = (data['Hydro_Gen'] + data['Solar_Gen']) / data
    ['Total_Gen'] * 100
data['Fossil_Pct'] = (data['Coal_Gen'] + data['Gas_Gen']) / data['
    Total_Gen'] * 100

# Save the enhanced feature dataset
data.to_csv("enhanced_features.csv", index=False)
```

Python Code for Visualization

Listing 6.3: 03_Visualisation.py

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Load data
df = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    CSV_files\featue_engg_data.csv")

# Convert 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'])

# Set date as index for resampling
df.set_index('Date', inplace=True)
weekly_demand = df['Demand'].resample('W').mean()

# Weekly average electricity demand plot
plt.figure(figsize=(14, 5))
plt.plot(weekly_demand.index, weekly_demand.values, color='royalblue',
     linewidth=2)
plt.title("Weekly Average Electricity Demand Over Time", fontsize=14)
plt.xlabel("Date", fontsize=12)
plt.ylabel("Demand (MWh)", fontsize=12)
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()

df.reset_index(inplace=True)
df1 = df.copy()

# Line plot for Demand, Net Generation, and CO2 Emissions
df1['Date'] = pd.to_datetime(df1['Date'], errors='coerce')
df1.sort_values('Date', inplace=True)
df1.set_index('Date', inplace=True)

plt.figure(figsize=(14, 6))
plt.plot(df1.index, df1['Demand'], label='Electricity Demand', alpha
    =0.7)
plt.plot(df1.index, df1['Net_Generation'], label='Net Generation',
    alpha=0.7)
plt.plot(df1.index, df1['CO2_Total_Emissions'], label=' C O  Emissions
    ', alpha=0.7)
plt.title('Electricity Demand, Net Generation, and  C O  Emissions
    Over Time')
plt.xlabel('Date')
plt.ylabel('Value (MWh or Tons)')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

# Monthly stacked area chart of energy mix
df1['Month'] = df1.index.to_period('M')
monthly_mix = df1.groupby('Month')[['Coal_Gen', 'Gas_Gen', '
    Nuclear_Gen', 'Hydro_Gen', 'Solar_Gen']].mean()
monthly_mix.index = monthly_mix.index.to_timestamp()

plt.figure(figsize=(14, 7))
plt.stackplot(monthly_mix.index,
              monthly_mix['Coal_Gen'],
              monthly_mix['Gas_Gen'],
              monthly_mix['Nuclear_Gen'],
```

```
                monthly_mix['Hydro_Gen'],
                monthly_mix['Solar_Gen'],
                labels=['Coal', 'Gas', 'Nuclear', 'Hydro', 'Solar'],
                alpha=0.85)
plt.title("Monthly Average Energy Generation by Source")
plt.xlabel("Month")
plt.ylabel("Generation (MWh)")
plt.legend(loc='upper left')
plt.grid(True)
plt.tight_layout()
plt.show()

# Pie chart for total emissions by source
total_emissions = {
    'Coal': df['CO2_Emissions_Coal'].sum(),
    'Gas': df['CO2_Emissions_Gas'].sum(),
    'Other Sources': df['CO2_Total_Emissions'].sum() - (df['
       CO2_Emissions_Coal'].sum() + df['CO2_Emissions_Gas'].sum())
}
custom_colors = ['#8B0000', '#e377c2', '#ffbf00']

plt.figure(figsize=(8, 8))
plt.pie(total_emissions.values(),
        labels=total_emissions.keys(),
        autopct='%1.1f%%',
        startangle=140,
        colors=custom_colors)
plt.title(' C O  Emissions Contribution by Source (National Level)')
plt.tight_layout()
plt.show()

# Monthly energy mix by region
df['Date'] = pd.to_datetime(df['Date'], dayfirst=True, errors='coerce'
   )
df['Month'] = df['Date'].dt.to_period('M')
regions = df['Region'].unique()
energy_sources = ['Coal_Gen', 'Gas_Gen', 'Nuclear_Gen', 'Hydro_Gen', '
   Solar_Gen']
energy_labels = ['Coal', 'Natural Gas', 'Nuclear', 'Hydro', 'Solar']

def get_monthly_mix(df, region):
    df_region = df[df['Region'] == region]
    monthly_mix = df_region.groupby('Month')[energy_sources].sum()
    monthly_mix.index = monthly_mix.index.to_timestamp()
    return monthly_mix

n_regions = len(regions)
n_cols = 3
n_rows = -(-n_regions // n_cols)

fig, axes = plt.subplots(n_rows, n_cols, figsize=(18, 5 * n_rows),
   sharex=True)
axes = axes.flatten()

for i, region in enumerate(regions):
    mix = get_monthly_mix(df, region)
    axes[i].stackplot(mix.index,
                      [mix[col] for col in energy_sources],
```

```
                        labels=energy_labels,
                        alpha=0.85)
    axes[i].set_title(f'Monthly Energy Mix - {region}')
    axes[i].set_ylabel("MWh")
    axes[i].grid(True)
    if i >= (n_rows - 1) * n_cols:
        axes[i].set_xlabel("Month")
    if i == 0:
        axes[i].legend(loc='upper left')

for j in range(i + 1, len(axes)):
    fig.delaxes(axes[j])

plt.tight_layout()
plt.show()

# Replace DayOfWeek numbers with names
day_map = {0: 'Monday', 1: 'Tuesday', 2: 'Wednesday', 3: 'Thursday',
           4: 'Friday', 5: 'Saturday', 6: 'Sunday'}
df['DayOfWeek'] = df['DayOfWeek'].map(day_map)

# Violin plot
plt.figure(figsize=(12, 6))
sns.violinplot(x='DayOfWeek', y='Demand', data=df,
               order=['Monday', 'Tuesday', 'Wednesday', 'Thursday', '
                  Friday', 'Saturday', 'Sunday'],
               inner='box', palette='Set2')
plt.title("Electricity Demand Distribution by Day of Week (Violin Plot
   )")
plt.xlabel("Day of Week")
plt.ylabel("Demand (MWh)")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Line plot by hour
avg_demand_by_hour = df.groupby('Hour')['Demand'].mean().reset_index()
plt.figure(figsize=(12, 6))
plt.plot(avg_demand_by_hour['Hour'], avg_demand_by_hour['Demand'],
   marker='o', linestyle='-', color='teal')
plt.title("Average Electricity Demand by Hour of Day (Line Plot)")
plt.xlabel("Hour")
plt.ylabel("Average Demand (MWh)")
plt.grid(True)
plt.tight_layout()
plt.show()

# Heatmap: Hour vs Day
heatmap1 = df.pivot_table(index='Hour', columns='DayOfWeek', values='
   Demand', aggfunc='mean')
heatmap1 = heatmap1[['Monday', 'Tuesday', 'Wednesday', 'Thursday', '
   Friday', 'Saturday', 'Sunday']]
plt.figure(figsize=(10, 8))
sns.heatmap(heatmap1, cmap='YlGnBu', linewidths=0.3)
plt.title("Average Electricity Demand: Hour vs Day of Week")
plt.xlabel("Day of Week")
plt.ylabel("Hour")
plt.tight_layout()
```

```
plt.show()

# Heatmap: Hour vs Month
heatmap2 = df.pivot_table(index='Hour', columns='Month', values='
    Demand', aggfunc='mean')
plt.figure(figsize=(12, 8))
sns.heatmap(heatmap2, cmap='YlOrRd', linewidths=0.3)
plt.title("Average Electricity Demand: Hour vs Month")
plt.xlabel("Month")
plt.ylabel("Hour")
plt.tight_layout()
plt.show()

# Region-level total comparison
region_grouped = df.groupby('Region').agg({
    'Demand': 'sum',
    'CO2_Total_Emissions': 'sum'
}).reset_index()

fig, ax1 = plt.subplots(figsize=(12, 6))
ax1.set_xlabel('Region')
ax1.set_ylabel('Total Electricity Demand', color='tab:green')
ax1.bar(region_grouped['Region'], region_grouped['Demand'], color='tab
    :green', alpha=0.6)
ax1.tick_params(axis='y', labelcolor='tab:green')

ax2 = ax1.twinx()
ax2.set_ylabel('Total  C O  Emissions', color='tab:red')
ax2.plot(region_grouped['Region'], region_grouped['CO2_Total_Emissions
    '], color='tab:red', marker='o')
ax2.tick_params(axis='y', labelcolor='tab:red')

plt.title('Total Electricity Demand and  C O  Emissions by Region')
fig.tight_layout()
plt.show()

# Region-specific demand heatmaps
for region in ['CAL', 'TEX']:
    regional_data = df[df['Region'] == region]
    heatmap = regional_data.pivot_table(index='Hour', columns='Month',
        values='Demand', aggfunc='mean')
    plt.figure(figsize=(12, 8))
    sns.heatmap(heatmap, cmap='YlGnBu' if region == 'TEX' else 'YlOrRd
        ', linewidths=0.3)
    plt.title(f"{region}     Average Electricity Demand (Hour vs Month
        )")
    plt.xlabel("Month")
    plt.ylabel("Hour")
    plt.tight_layout()
    plt.show()

# Renewable vs Emissions Trend - California
region_df = df[df['Region'] == 'CAL'].copy()
region_df['Date'] = pd.to_datetime(region_df['Date'])
daily_trend = region_df.groupby('Date').agg({
    'Renewable_Pct': 'mean',
    'CO2_Total_Emissions': 'sum'
}).reset_index()
```

```
fig, ax1 = plt.subplots(figsize=(14, 6))
ax1.set_title("Historical Trend: Renewable Energy Share vs  C O
    Emissions (California)")
ax1.plot(daily_trend['Date'], daily_trend['Renewable_Pct'], color='
    green')
ax1.set_xlabel("Date")
ax1.set_ylabel("Renewable Energy Share (%)", color='green')
ax1.tick_params(axis='y', labelcolor='green')

ax2 = ax1.twinx()
ax2.plot(daily_trend['Date'], daily_trend['CO2_Total_Emissions'],
    color='red', alpha=0.6)
ax2.set_ylabel(" C O  Emissions (tons)", color='red')
ax2.tick_params(axis='y', labelcolor='red')
plt.grid(True)
plt.tight_layout()
plt.show()

# Renewable vs Emissions Trend - Texas
region_df = df[df['Region'] == 'TEX'].copy()
region_df['Date'] = pd.to_datetime(region_df['Date'])
daily_trend = region_df.groupby('Date').agg({
    'Renewable_Pct': 'mean',
    'CO2_Total_Emissions': 'sum'
}).reset_index()

fig, ax1 = plt.subplots(figsize=(14, 6))
ax1.set_title("Historical Trend: Renewable Energy Share vs  C O
    Emissions (Texas)")
ax1.plot(daily_trend['Date'], daily_trend['Renewable_Pct'], color='
    green')
ax1.set_xlabel("Date")
ax1.set_ylabel("Renewable Energy Share (%)", color='green')
ax1.tick_params(axis='y', labelcolor='green')

ax2 = ax1.twinx()
ax2.plot(daily_trend['Date'], daily_trend['CO2_Total_Emissions'],
    color='red', alpha=0.6)
ax2.set_ylabel(" C O  Emissions (tons)", color='red')
ax2.tick_params(axis='y', labelcolor='red')
plt.grid(True)
plt.tight_layout()
plt.show()
```

Python Code for Modelling

Listing 6.4: 04_Modelling.py

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
import tensorflow as tf
```

```
df = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    CSV_files\featue_engg_data1.csv")
df['Date'] = pd.to_datetime(df['Date'])
df

df.dtypes

# Load and prepare data
#df = pd.read_csv("featue_engg_data1.csv")
df["Date"] = pd.to_datetime(df["Date"])
df.set_index("Date", inplace=True)

# Group by date for national data (summing values across regions per
    day)
national = df.groupby(df.index).agg({
    'Demand': 'sum',
    'CO2_Total_Emissions': 'sum',
    'Hour': 'mean',
    'Month': 'mean',
    'DayOfWeek': 'mean',
    'Is_Weekend': 'mean',
    'DayOfYear': 'mean',
    'WeekOfYear': 'mean',
    'Season_Autumn': 'mean',
    'Season_Spring': 'mean',
    'Season_Summer': 'mean',
    'Season_Winter': 'mean',
    'Renewable_Pct': 'mean',
    'Fossil_Pct': 'mean',
    'Demand_Prev_Hour': 'mean',
    'Demand_Yesterday_Same_Hour': 'mean',
    'Demand_Last_Week_Same_Hour': 'mean',
    'Rolling_Mean_3H': 'mean',
    'Rolling_Mean_24H': 'mean'
})

# Select features and targets
features = national[[
    'Hour', 'Month', 'DayOfWeek', 'Is_Weekend', 'DayOfYear', '
       WeekOfYear',
    'Season_Autumn', 'Season_Spring', 'Season_Summer', 'Season_Winter'
       ,
    'Renewable_Pct', 'Fossil_Pct',
    'Demand_Prev_Hour', 'Demand_Yesterday_Same_Hour', '
       Demand_Last_Week_Same_Hour',
    'Rolling_Mean_3H', 'Rolling_Mean_24H'
]]
targets = national[['Demand', 'CO2_Total_Emissions']]

# Scale both features and targets
scaler_X = MinMaxScaler()
scaler_y = MinMaxScaler()
X_scaled = scaler_X.fit_transform(features)
y_scaled = scaler_y.fit_transform(targets)

# Convert to supervised sequences
def create_sequences(X, y, lookback=30, forecast_horizon=90):
```

```
    Xs, ys = [], []
    for i in range(len(X) - lookback - forecast_horizon):
        Xs.append(X[i:i+lookback])
        ys.append(y[i+lookback:i+lookback+forecast_horizon])
    return np.array(Xs), np.array(ys)

X_seq, y_seq = create_sequences(X_scaled, y_scaled)

# Define LSTM model
model = Sequential([
    LSTM(64, input_shape=(X_seq.shape[1], X_seq.shape[2])),
    Dense(y_seq.shape[1] * y_seq.shape[2])  # output shape: (
       forecast_horizon * num_targets)
])
model.compile(optimizer='adam', loss='mse')
model.fit(X_seq, y_seq.reshape(y_seq.shape[0], -1), epochs=5,
   batch_size=32)

# Forecast next 90 days
X_input = X_scaled[-30:]
forecast_scaled = model.predict(X_input.reshape(1, 30, X_seq.shape[2])
   ).reshape(90, 2)
forecast = scaler_y.inverse_transform(forecast_scaled)


import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense


features = national[[
    'Hour', 'Month', 'DayOfWeek', 'Is_Weekend', 'DayOfYear', '
       WeekOfYear',
    'Season_Autumn', 'Season_Spring', 'Season_Summer', 'Season_Winter'
       ,
    'Renewable_Pct', 'Fossil_Pct',
    'Demand_Prev_Hour', 'Demand_Yesterday_Same_Hour', '
       Demand_Last_Week_Same_Hour',
    'Rolling_Mean_3H', 'Rolling_Mean_24H'
]]
targets = national[['Demand', 'CO2_Total_Emissions']]

 #Scale
scaler_X = MinMaxScaler()
scaler_y = MinMaxScaler()
X_scaled = scaler_X.fit_transform(features)
y_scaled = scaler_y.fit_transform(targets)

 #Create sequences
def create_sequences(X, y, lookback=30):
    Xs, ys, y_actual = [], [], []
    for i in range(lookback, len(X)):
        Xs.append(X[i-lookback:i])
        ys.append(y[i])
        y_actual.append(i)   save index for inverse mapping
    return np.array(Xs), np.array(ys), np.array(y_actual)
```

```
X_seq, y_seq, y_idx = create_sequences(X_scaled, y_scaled)

 ------------------------
 #Step 2: Train/Test Split
 ------------------------
split = int(0.8 * len(X_seq))
X_train, X_test = X_seq[:split], X_seq[split:]
y_train, y_test = y_seq[:split], y_seq[split:]
test_idx = y_idx[split:]

 ------------------------
 #Step 3: Train LSTM
 ------------------------
model = Sequential([
    LSTM(64, input_shape=(X_train.shape[1], X_train.shape[2])),
    Dense(2)   single-step prediction (2 outputs)
])
model.compile(optimizer='adam', loss='mse')
model.fit(X_train, y_train, epochs=10, batch_size=32)

 ------------------------
 #Step 4: Predict and Inverse Transform
 ------------------------
y_pred_scaled = model.predict(X_test)
y_pred = scaler_y.inverse_transform(y_pred_scaled)
y_true = scaler_y.inverse_transform(y_test)

 ------------------------
 #Step 5: Plot Predictions vs Actuals
 ------------------------
dates = national.index[test_idx]

plt.figure(figsize=(14, 5))
plt.plot(dates, y_true[:, 0], label='Actual Demand')
plt.plot(dates, y_pred[:, 0], label='Predicted Demand')
plt.title('Electricity Demand: Actual vs Predicted')
plt.legend()
plt.grid(True)
plt.show()

plt.figure(figsize=(14, 5))
plt.plot(dates, y_true[:, 1], label='Actual  C O  Emissions')
plt.plot(dates, y_pred[:, 1], label='Predicted  C O  Emissions')
plt.title(' C O  Emissions: Actual vs Predicted')
plt.legend()
plt.grid(True)
plt.show()

from sklearn.metrics import mean_squared_error, mean_absolute_error
import numpy as np

 #Define Mean Absolute Percentage Error
def mean_absolute_percentage_error(y_true, y_pred):
    y_true, y_pred = np.array(y_true), np.array(y_pred)
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100

  #Demand Metrics
```

```
rmse_demand = np.sqrt(mean_squared_error(y_true[:, 0], y_pred[:, 0]))
mae_demand = mean_absolute_error(y_true[:, 0], y_pred[:, 0])
mape_demand = mean_absolute_percentage_error(y_true[:, 0], y_pred[:,
    0])

 # C O   Metrics
rmse_co2 = np.sqrt(mean_squared_error(y_true[:, 1], y_pred[:, 1]))
mae_co2 = mean_absolute_error(y_true[:, 1], y_pred[:, 1])
mape_co2 = mean_absolute_percentage_error(y_true[:, 1], y_pred[:, 1])

print("    Electricity Demand Forecast Accuracy:")
print(f"RMSE: {rmse_demand:.2f}, MAE: {mae_demand:.2f}, MAPE: {
    mape_demand:.2f}%")

print("\ n    C O   Emissions Forecast Accuracy:")
print(f"RMSE: {rmse_co2:.2f}, MAE: {mae_co2:.2f}, MAPE: {mape_co2:.2f
    }%")


df = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
    CSV_files\featue_engg_data1.csv")
df['Date'] = pd.to_datetime(df['Date'])
df

from sklearn.metrics import mean_absolute_error, mean_squared_error
import numpy as np
import matplotlib.pyplot as plt

def mean_absolute_percentage_error(y_true, y_pred):
    """Calculate Mean Absolute Percentage Error (MAPE)"""
    y_true, y_pred = np.array(y_true), np.array(y_pred)
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100

def eval_region_prediction(region_code, display_name=None):
    print(f"\ n     Evaluating prediction accuracy for {region_code
        }...")

    region_df = df[df['Region'] == region_code].copy()
    if region_df.empty:
        print(f"       No data found for region: {region_code}")
        return

    region_df['Date'] = pd.to_datetime(region_df['Date'])
    region_df.set_index('Date', inplace=True)

    daily_data = region_df.groupby(region_df.index).agg({
        'Demand': 'sum',
        'CO2_Total_Emissions': 'sum',
        'Hour': 'mean',
        'Month': 'mean',
        'DayOfWeek': 'mean',
        'Is_Weekend': 'mean',
        'DayOfYear': 'mean',
        'WeekOfYear': 'mean',
        'Season_Autumn': 'mean',
        'Season_Spring': 'mean',
        'Season_Summer': 'mean',
        'Season_Winter': 'mean',
```

```
    'Renewable_Pct': 'mean',
    'Fossil_Pct': 'mean',
    'Demand_Prev_Hour': 'mean',
    'Demand_Yesterday_Same_Hour': 'mean',
    'Demand_Last_Week_Same_Hour': 'mean',
    'Rolling_Mean_3H': 'mean',
    'Rolling_Mean_24H': 'mean'
})

features = daily_data[[
    'Hour', 'Month', 'DayOfWeek', 'Is_Weekend', 'DayOfYear', '
       WeekOfYear',
    'Season_Autumn', 'Season_Spring', 'Season_Summer', '
       Season_Winter',
    'Renewable_Pct', 'Fossil_Pct',
    'Demand_Prev_Hour', 'Demand_Yesterday_Same_Hour', '
       Demand_Last_Week_Same_Hour',
    'Rolling_Mean_3H', 'Rolling_Mean_24H'
]]
targets = daily_data[['Demand', 'CO2_Total_Emissions']]

 Scale
scaler_X = MinMaxScaler()
scaler_y = MinMaxScaler()
X_scaled = scaler_X.fit_transform(features)
y_scaled = scaler_y.fit_transform(targets)

def create_sequences(X, y, lookback=30):
    Xs, ys, idxs = [], [], []
    for i in range(lookback, len(X)):
        Xs.append(X[i-lookback:i])
        ys.append(y[i])
        idxs.append(i)
    return np.array(Xs), np.array(ys), np.array(idxs)

X_seq, y_seq, idx_seq = create_sequences(X_scaled, y_scaled)

split = int(0.8 * len(X_seq))
X_train, X_test = X_seq[:split], X_seq[split:]
y_train, y_test = y_seq[:split], y_seq[split:]
test_idx = idx_seq[split:]

model = Sequential([
    LSTM(64, input_shape=(X_train.shape[1], X_train.shape[2])),
    Dense(2)
])
model.compile(optimizer='adam', loss='mse')
model.fit(X_train, y_train, epochs=10, batch_size=32, verbose=0)

 Predict
y_pred_scaled = model.predict(X_test)
y_pred = scaler_y.inverse_transform(y_pred_scaled)
y_true = scaler_y.inverse_transform(y_test)
test_dates = daily_data.index[test_idx]
title_name = display_name if display_name else region_code

      Accuracy metrics
mae_demand = mean_absolute_error(y_true[:, 0], y_pred[:, 0])
```

```
    rmse_demand = np.sqrt(mean_squared_error(y_true[:, 0], y_pred[:,
        0]))
    mape_demand = mean_absolute_percentage_error(y_true[:, 0], y_pred
        [:, 0])

    mae_co2 = mean_absolute_error(y_true[:, 1], y_pred[:, 1])
    rmse_co2 = np.sqrt(mean_squared_error(y_true[:, 1], y_pred[:, 1]))
    mape_co2 = mean_absolute_percentage_error(y_true[:, 1], y_pred[:,
        1])

    print(f"--- {title_name} ---")
    print(f"Electricity Demand     MAE: {mae_demand:.2f}, RMSE: {
        rmse_demand:.2f}, MAPE: {mape_demand:.2f}%")
    print(f" C O   Emissions          MAE: {mae_co2:.2f}, RMSE: {
        rmse_co2:.2f}, MAPE: {mape_co2:.2f}%\n")

     Plot Demand
    plt.figure(figsize=(14, 5))
    plt.plot(test_dates, y_true[:, 0], label='Actual Demand')
    plt.plot(test_dates, y_pred[:, 0], label='Predicted Demand')
    plt.title(f"{title_name}     Electricity Demand\nMAE: {mae_demand
        :.2f}, RMSE: {rmse_demand:.2f}, MAPE: {mape_demand:.2f}%")
    plt.legend()
    plt.grid(True)
    plt.show()

     Plot  C O
    plt.figure(figsize=(14, 5))
    plt.plot(test_dates, y_true[:, 1], label='Actual  C O  Emissions')
    plt.plot(test_dates, y_pred[:, 1], label='Predicted  C O
        Emissions')
    plt.title(f"{title_name}      C O  Emissions\nMAE: {mae_co2:.2f},
        RMSE: {rmse_co2:.2f}, MAPE: {mape_co2:.2f}%")
    plt.legend()
    plt.grid(True)
    plt.show()

    return y_true, y_pred, y_test, test_idx, daily_data, {
    'Region': display_name or region_code,
    'Demand': {
        'MAE': mae_demand,
        'RMSE': rmse_demand,
        'MAPE': mape_demand
    },
    'CO2': {
        'MAE': mae_co2,
        'RMSE': rmse_co2,
        'MAPE': mape_co2
        }
    }


y_true, y_pred, y_test, test_idx, daily_data, cal_metrics =
    eval_region_prediction("CAL", "California")
y_true, y_pred, y_test, test_idx, daily_data, tex_metrics =
    eval_region_prediction("TEX", "Texas")
```

```
def simulate_renewable_scenarios(region_code, display_name=None):
    import matplotlib.pyplot as plt
    import seaborn as sns
    from sklearn.preprocessing import MinMaxScaler
    import numpy as np
    import pandas as pd
    from tensorflow.keras.models import Sequential
    from tensorflow.keras.layers import LSTM, Dense
    import tensorflow as tf

    print(f"\ n     Simulating renewable energy scenarios for {
        region_code}...")
     Set seeds for reproducibility
    np.random.seed(42)
    tf.random.set_seed(42)
    lookback = 15   LSTM rolling window

     1       Data Preparation
    region_df = df[df['Region'] == region_code].copy()
    region_df['Date'] = pd.to_datetime(region_df['Date'])
    region_df.set_index('Date', inplace=True)

     Aggregate daily
    daily_data = region_df.groupby(region_df.index).agg({
        'Demand': 'sum',
        'CO2_Total_Emissions': 'sum',
        'Hour': 'mean',
        'Month': 'mean',
        'DayOfWeek': 'mean',
        'Is_Weekend': 'mean',
        'DayOfYear': 'mean',
        'WeekOfYear': 'mean',
        'Season_Autumn': 'mean',
        'Season_Spring': 'mean',
        'Season_Summer': 'mean',
        'Season_Winter': 'mean',
        'Renewable_Pct': 'mean',
        'Fossil_Pct': 'mean',
        'Demand_Prev_Hour': 'mean',
        'Demand_Yesterday_Same_Hour': 'mean',
        'Demand_Last_Week_Same_Hour': 'mean',
        'Rolling_Mean_3H': 'mean',
        'Rolling_Mean_24H': 'mean'
    })

    features = daily_data[[
        'Hour', 'Month', 'DayOfWeek', 'Is_Weekend', 'DayOfYear', '
           WeekOfYear',
        'Season_Autumn', 'Season_Spring', 'Season_Summer', '
           Season_Winter',
        'Renewable_Pct', 'Fossil_Pct',
        'Demand_Prev_Hour', 'Demand_Yesterday_Same_Hour', '
           Demand_Last_Week_Same_Hour',
        'Rolling_Mean_3H', 'Rolling_Mean_24H'
    ]]
    targets = daily_data[['Demand', 'CO2_Total_Emissions']]

     Scale
```

```
scaler_X = MinMaxScaler()
scaler_y = MinMaxScaler()
X_scaled = scaler_X.fit_transform(features)
y_scaled = scaler_y.fit_transform(targets)

max_rows = 20000
X_scaled = X_scaled[-max_rows:]
y_scaled = y_scaled[-max_rows:]
dates_all = daily_data.index[-max_rows:]

 Sequences
def create_sequences(X, y, dates, lookback):
    Xs, ys, idxs = [], [], []
    for i in range(lookback, len(X)):
        Xs.append(X[i - lookback:i])
        ys.append(y[i])
        idxs.append(dates[i])
    return np.array(Xs), np.array(ys), idxs

X_seq, y_seq, idx_seq = create_sequences(X_scaled, y_scaled,
   dates_all, lookback)

 2       Train LSTM
model = Sequential([
    LSTM(64, input_shape=(X_seq.shape[1], X_seq.shape[2])),
    Dense(2)
])
model.compile(optimizer='adam', loss='mse')
model.fit(X_seq, y_seq, epochs=10, batch_size=32, verbose=0)

 3       Simulate FUTURE (2023  2024  )
future_dates = pd.date_range(start=daily_data.index[-1] + pd.
   Timedelta(days=1), periods=365, freq='D')

base_df = pd.DataFrame({
    'Date': future_dates,
    'Hour': 12,
    'Month': future_dates.month,
    'DayOfWeek': future_dates.dayofweek,
    'Is_Weekend': future_dates.dayofweek.isin([5, 6]).astype(int),
    'DayOfYear': future_dates.dayofyear,
    'WeekOfYear': future_dates.isocalendar().week,
    'Season_Winter': ((future_dates.month <= 2) | (future_dates.
       month == 12)).astype(int),
    'Season_Spring': future_dates.month.isin([3, 4, 5]).astype(int
       ),
    'Season_Summer': future_dates.month.isin([6, 7, 8]).astype(int
       ),
    'Season_Autumn': future_dates.month.isin([9, 10, 11]).astype(
       int),
})

for col in ['Demand_Prev_Hour', 'Demand_Yesterday_Same_Hour', '
   Demand_Last_Week_Same_Hour',
            'Rolling_Mean_3H', 'Rolling_Mean_24H']:
    base_df[col] = region_df[col].mean()

renew_base = region_df['Renewable_Pct'].mean()
```

```
fossil_base = region_df['Fossil_Pct'].mean()

scenarios = {
    "Base": (renew_base, fossil_base),
    "+10% Renewables": (min(renew_base + 10, 100), max(fossil_base
        - 10, 0)),
    "+20% Renewables": (min(renew_base + 20, 100), max(fossil_base
        - 20, 0)),
    "+30% Renewables": (min(renew_base + 30, 100), max(fossil_base
        - 30, 0)),
}

predictions_future = {}
for name, (renew, fossil) in scenarios.items():
    df_copy = base_df.copy()
    df_copy['Renewable_Pct'] = renew
    df_copy['Fossil_Pct'] = fossil
    df_copy = df_copy[features.columns]
    scaled = scaler_X.transform(df_copy)

    input_seq = X_scaled[-lookback:].copy()
    forecast_scaled = []
    for i in range(365):
        input_seq = np.vstack([input_seq[1:], scaled[i]])
        pred = model.predict(input_seq.reshape(1, lookback,
            X_scaled.shape[1]), verbose=0)[0]
        forecast_scaled.append(pred)
    forecast = scaler_y.inverse_transform(forecast_scaled)
    predictions_future[name] = forecast[:, 1]

 4       Historical simulation with changed renewables
recent_df = features.tail(365).copy()
base_emissions_hist = scaler_y.inverse_transform(y_scaled[-365:])
    [:, 1]
predictions_hist = {}

for name, (renew, fossil) in scenarios.items():
    mod_df = recent_df.copy()
    mod_df['Renewable_Pct'] = renew
    mod_df['Fossil_Pct'] = fossil
    scaled_mod = scaler_X.transform(mod_df)
    input_seq = X_scaled[-(365 + lookback):-365]   use prior
        window

    forecast_scaled = []
    for i in range(365):
        input_seq = np.vstack([input_seq[1:], scaled_mod[i]])
        pred = model.predict(input_seq.reshape(1, lookback,
            X_scaled.shape[1]), verbose=0)[0]
        forecast_scaled.append(pred)
    forecast = scaler_y.inverse_transform(forecast_scaled)
    predictions_hist[name] = forecast[:, 1]

 5       Plot Historical Impact
plt.figure(figsize=(14, 5))
plt.plot(dates_all[-365:], base_emissions_hist, label='Actual CO2
    Emissions', color='black')
plt.plot(dates_all[-365:], predictions_hist["+30% Renewables"],
```

```
        label='Simulated CO2 (+30% Renewables)', color='green')
    plt.title(f"{display_name or region_code}: Historical Simulation
        of CO2 Emissions with Renewables")
    plt.xlabel("Date")
    plt.ylabel("CO2 Emissions (tons)")
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plt.show()

     6       Plot Future Scenario
    base_scaled = MinMaxScaler().fit_transform(predictions_future["
        Base"].reshape(-1, 1)).flatten()
    sim_scaled = MinMaxScaler().fit_transform(predictions_future["+30%
         Renewables"].reshape(-1, 1)).flatten()

    plt.figure(figsize=(14, 5))
    plt.plot(future_dates, base_scaled, label="Original CO2 Emissions"
        , color='orange')
    plt.plot(future_dates, sim_scaled, label="Simulated CO2 (30% more
        renewables)", color='green')
    plt.title(f"{display_name or region_code}: Future CO2 Emissions
        Forecast with 30% Renewables")
    plt.xlabel("Date")
    plt.ylabel("CO2 Emissions (Scaled)")
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plt.show()

    return {
        'Region': display_name or region_code,
        'Historical_Base_CO2': np.sum(base_emissions_hist),
        'Historical_30%Renew_CO2': np.sum(predictions_hist["+30%
            Renewables"]),
        'Future_Base_CO2': np.sum(predictions_future["Base"]),
        'Future_30%Renew_CO2': np.sum(predictions_future["+30%
            Renewables"])
    }


cal_results = simulate_renewable_scenarios("CAL", "California")
tex_results = simulate_renewable_scenarios("TEX", "Texas")

if isinstance(cal_results, dict):
    cal_results = [cal_results]
if isinstance(tex_results, dict):
    tex_results = [tex_results]

results_df = pd.DataFrame(cal_results + tex_results)


 Calculate  C O  reduction percent for historical and future
results_df['Historical_CO2_Reduction_Percent'] = (
    (results_df['Historical_Base_CO2'] - results_df['Historical_30%
        Renew_CO2']) / results_df['Historical_Base_CO2']
) * 100
```

```
results_df['Future_CO2_Reduction_Percent'] = (
    (results_df['Future_Base_CO2'] - results_df['Future_30%Renew_CO2'
       ]) / results_df['Future_Base_CO2']
) * 100



import matplotlib.pyplot as plt
import seaborn as sns

 Melt for easier plotting
melted_df = results_df.melt(
    id_vars='Region',
    value_vars=['Historical_CO2_Reduction_Percent', '
       Future_CO2_Reduction_Percent'],
    var_name='Scenario_Type',
    value_name='Reduction_Percent'
)

 Rename for clarity
melted_df['Scenario_Type'] = melted_df['Scenario_Type'].replace({
    'Historical_CO2_Reduction_Percent': 'Historical (30% Renewables)',
    'Future_CO2_Reduction_Percent': 'Future (30% Renewables)'
})

 Plot
plt.figure(figsize=(10, 6))
sns.barplot(
    data=melted_df,
    x='Reduction_Percent',
    y='Region',
    hue='Scenario_Type',
    palette='viridis'
)

plt.title("CO2 Reduction with 30% More Renewables (Historical vs.
   Future)", fontsize=14)
plt.xlabel("CO2 Reduction (%)")
plt.ylabel("Region")
plt.legend(title="Scenario Type", loc='lower right')
plt.xlim(0, melted_df['Reduction_Percent'].max() + 10)
plt.tight_layout()
plt.show()

summary = results_df[['Region', 'Historical_CO2_Reduction_Percent', '
   Future_CO2_Reduction_Percent']].copy()
summary['Historical_CO2_Reduction_Percent'] = summary['
   Historical_CO2_Reduction_Percent'].map('{:.2f}%'.format)
summary['Future_CO2_Reduction_Percent'] = summary['
   Future_CO2_Reduction_Percent'].map('{:.2f}%'.format)

print(summary.to_string(index=False))


df = pd.read_csv(r"C:\Users\rosem\Downloads\FINAL_PROJECT_WORKS\
   CSV_files\featue_engg_data1.csv")
df['Date'] = pd.to_datetime(df['Date'])
df
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import SimpleRNN, Dense


region_df = df[df['Region'] == 'CAL'].copy()   Example for California
region_df['Date'] = pd.to_datetime(region_df['Date'])
region_df.set_index('Date', inplace=True)

 Daily aggregation
daily_data = region_df.groupby(region_df.index).agg({
    'Demand': 'sum',
    'CO2_Total_Emissions': 'sum',
    'Hour': 'mean',
    'Month': 'mean',
    'DayOfWeek': 'mean',
    'Is_Weekend': 'mean',
    'DayOfYear': 'mean',
    'WeekOfYear': 'mean',
    'Season_Autumn': 'mean',
    'Season_Spring': 'mean',
    'Season_Summer': 'mean',
    'Season_Winter': 'mean',
    'Renewable_Pct': 'mean',
    'Fossil_Pct': 'mean',
    'Demand_Prev_Hour': 'mean',
    'Demand_Yesterday_Same_Hour': 'mean',
    'Demand_Last_Week_Same_Hour': 'mean',
    'Rolling_Mean_3H': 'mean',
    'Rolling_Mean_24H': 'mean'
})


features = daily_data[[   Same features
    'Hour', 'Month', 'DayOfWeek', 'Is_Weekend', 'DayOfYear', '
       WeekOfYear',
    'Season_Autumn', 'Season_Spring', 'Season_Summer', 'Season_Winter'
       ,
    'Renewable_Pct', 'Fossil_Pct',
    'Demand_Prev_Hour', 'Demand_Yesterday_Same_Hour', '
       Demand_Last_Week_Same_Hour',
    'Rolling_Mean_3H', 'Rolling_Mean_24H'
]]
targets = daily_data[['Demand', 'CO2_Total_Emissions']]

scaler_X = MinMaxScaler()
scaler_y = MinMaxScaler()

X_scaled = scaler_X.fit_transform(features)
y_scaled = scaler_y.fit_transform(targets)


def create_sequences(X, y, lookback=30):
```

```
    Xs, ys = [], []
    for i in range(lookback, len(X)):
        Xs.append(X[i-lookback:i])
        ys.append(y[i])
    return np.array(Xs), np.array(ys)

X_seq, y_seq = create_sequences(X_scaled, y_scaled)


split = int(0.8 * len(X_seq))
X_train, X_test = X_seq[:split], X_seq[split:]
y_train, y_test = y_seq[:split], y_seq[split:]


model = Sequential([
    SimpleRNN(64, input_shape=(X_train.shape[1], X_train.shape[2])),
         RNN Layer
    Dense(2)   2 outputs: Demand, CO2
])
model.compile(optimizer='adam', loss='mse')
model.fit(X_train, y_train, epochs=10, batch_size=32, verbose=1)


y_pred_scaled = model.predict(X_test)
y_pred = scaler_y.inverse_transform(y_pred_scaled)
y_true = scaler_y.inverse_transform(y_test)


test_dates = daily_data.index[30 + split:]   Adjusted for lookback
    offset

rnn_results_df = pd.DataFrame({
    'Date': test_dates,
    'Region': 'California',
    'Model': 'RNN',
    'Actual_Demand': y_true[:, 0],
    'Predicted_Demand': y_pred[:, 0],
    'Actual_CO2': y_true[:, 1],
    'Predicted_CO2': y_pred[:, 1]
})

rnn_results_df.to_csv('rnn_california_forecast.csv', index=False)

import matplotlib.pyplot as plt

plt.figure(figsize=(14, 5))
plt.plot(test_dates, y_true[:, 0], label='Actual Demand', color='blue'
    , linewidth=2)
plt.plot(test_dates, y_pred[:, 0], label='Predicted Demand (RNN)',
    color='orange', linestyle='--')
plt.title('California     Electricity Demand Forecast (Actual vs
    Predicted) using RNN')
plt.xlabel('Date')
plt.ylabel('Electricity Demand (MWh)')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
plt.figure(figsize=(14, 5))
plt.plot(test_dates, y_true[:, 1], label='Actual  C O  Emissions',
    color='green', linewidth=2)
plt.plot(test_dates, y_pred[:, 1], label='Predicted  C O  Emissions (
    RNN)', color='red', linestyle='--')
plt.title('California      C O  Emissions Forecast (Actual vs
    Predicted) using RNN')
plt.xlabel('Date')
plt.ylabel(' C O  Emissions (metric tons)')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

region_df_tex = df[df['Region'] == 'TEX'].copy()   Example for
    California
region_df_tex['Date'] = pd.to_datetime(region_df_tex['Date'])
region_df_tex.set_index('Date', inplace=True)

 Daily aggregation
daily_data = region_df_tex.groupby(region_df_tex.index).agg({
    'Demand': 'sum',
    'CO2_Total_Emissions': 'sum',
    'Hour': 'mean',
    'Month': 'mean',
    'DayOfWeek': 'mean',
    'Is_Weekend': 'mean',
    'DayOfYear': 'mean',
    'WeekOfYear': 'mean',
    'Season_Autumn': 'mean',
    'Season_Spring': 'mean',
    'Season_Summer': 'mean',
    'Season_Winter': 'mean',
    'Renewable_Pct': 'mean',
    'Fossil_Pct': 'mean',
    'Demand_Prev_Hour': 'mean',
    'Demand_Yesterday_Same_Hour': 'mean',
    'Demand_Last_Week_Same_Hour': 'mean',
    'Rolling_Mean_3H': 'mean',
    'Rolling_Mean_24H': 'mean'
})


features = daily_data[[   Same features
    'Hour', 'Month', 'DayOfWeek', 'Is_Weekend', 'DayOfYear', '
       WeekOfYear',
    'Season_Autumn', 'Season_Spring', 'Season_Summer', 'Season_Winter'
       ,
    'Renewable_Pct', 'Fossil_Pct',
    'Demand_Prev_Hour', 'Demand_Yesterday_Same_Hour', '
       Demand_Last_Week_Same_Hour',
    'Rolling_Mean_3H', 'Rolling_Mean_24H'
]]
targets = daily_data[['Demand', 'CO2_Total_Emissions']]

scaler_X = MinMaxScaler()
scaler_y = MinMaxScaler()
```

```
X_scaled = scaler_X.fit_transform(features)
y_scaled = scaler_y.fit_transform(targets)


def create_sequences(X, y, lookback=30):
    Xs, ys = [], []
    for i in range(lookback, len(X)):
        Xs.append(X[i-lookback:i])
        ys.append(y[i])
    return np.array(Xs), np.array(ys)

X_seq, y_seq = create_sequences(X_scaled, y_scaled)


split = int(0.8 * len(X_seq))
X_train, X_test = X_seq[:split], X_seq[split:]
y_train, y_test = y_seq[:split], y_seq[split:]

model = Sequential([
    SimpleRNN(64, input_shape=(X_train.shape[1], X_train.shape[2])),
         RNN Layer
    Dense(2)   2 outputs: Demand, CO2
])
model.compile(optimizer='adam', loss='mse')
model.fit(X_train, y_train, epochs=10, batch_size=32, verbose=1)


y_pred_scaled = model.predict(X_test)
y_pred = scaler_y.inverse_transform(y_pred_scaled)
y_true = scaler_y.inverse_transform(y_test)

test_dates = daily_data.index[30 + split:]   Adjusted for lookback
   offset

rnn_results_df = pd.DataFrame({
    'Date': test_dates,
    'Region': 'Texas',
    'Model': 'RNN',
    'Actual_Demand': y_true[:, 0],
    'Predicted_Demand': y_pred[:, 0],
    'Actual_CO2': y_true[:, 1],
    'Predicted_CO2': y_pred[:, 1]
})

rnn_results_df.to_csv('rnn_Texas_forecast.csv', index=False)

import matplotlib.pyplot as plt

plt.figure(figsize=(14, 5))
plt.plot(test_dates, y_true[:, 0], label='Actual Demand', color='blue'
   , linewidth=2)
plt.plot(test_dates, y_pred[:, 0], label='Predicted Demand (RNN)',
   color='orange', linestyle='--')
plt.title('Texas     Electricity Demand Forecast (Actual vs Predicted)
    using RNN')
plt.xlabel('Date')
plt.ylabel('Electricity Demand (MWh)')
```

```
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

plt.figure(figsize=(14, 5))
plt.plot(test_dates, y_true[:, 1], label='Actual  C O  Emissions',
    color='green', linewidth=2)
plt.plot(test_dates, y_pred[:, 1], label='Predicted  C O  Emissions (
    RNN)', color='red', linestyle='--')
plt.title('Texas      C O  Emissions Forecast (Actual vs Predicted)
    using RNN')
plt.xlabel('Date')
plt.ylabel(' C O  Emissions (metric tons)')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.neural_network import BernoulliRBM
from sklearn.neural_network import MLPRegressor
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_absolute_error, mean_squared_error

 #Define evaluation function
def evaluate_forecast(y_true, y_pred):
    rmse = np.sqrt(mean_squared_error(y_true, y_pred))
    mae = mean_absolute_error(y_true, y_pred)
    mape = np.mean(np.abs((y_true - y_pred) / y_true)) * 100
    return rmse, mae, mape

def rbm_forecasting(region_code, display_name):
    print(f"\ n     Processing region: {display_name}")

     #Filter the data
    region_df = df[df['Region'] == region_code].copy()
    region_df['Date'] = pd.to_datetime(region_df['Date'])
    region_df.set_index('Date', inplace=True)

     #Daily aggregation
    daily_data = region_df.groupby(region_df.index).agg({
        'Demand': 'sum',
        'CO2_Total_Emissions': 'sum',
        'Hour': 'mean',
        'Month': 'mean',
        'DayOfWeek': 'mean',
        'Is_Weekend': 'mean',
        'DayOfYear': 'mean',
        'WeekOfYear': 'mean',
        'Season_Autumn': 'mean',
        'Season_Spring': 'mean',
        'Season_Summer': 'mean',
        'Season_Winter': 'mean',
```

```
    'Renewable_Pct': 'mean',
    'Fossil_Pct': 'mean',
    'Demand_Prev_Hour': 'mean',
    'Demand_Yesterday_Same_Hour': 'mean',
    'Demand_Last_Week_Same_Hour': 'mean',
    'Rolling_Mean_3H': 'mean',
    'Rolling_Mean_24H': 'mean'
})

features = daily_data[[
    'Hour', 'Month', 'DayOfWeek', 'Is_Weekend', 'DayOfYear', '
       WeekOfYear',
    'Season_Autumn', 'Season_Spring', 'Season_Summer', '
       Season_Winter',
    'Renewable_Pct', 'Fossil_Pct',
    'Demand_Prev_Hour', 'Demand_Yesterday_Same_Hour', '
       Demand_Last_Week_Same_Hour',
    'Rolling_Mean_3H', 'Rolling_Mean_24H'
]]
targets = daily_data[['Demand', 'CO2_Total_Emissions']]

 #Scaling
scaler_X = MinMaxScaler()
scaler_y = MinMaxScaler()
X_scaled = scaler_X.fit_transform(features)
y_scaled = scaler_y.fit_transform(targets)

 #Train/Test split
split = int(0.8 * len(X_scaled))
X_train, X_test = X_scaled[:split], X_scaled[split:]
y_train, y_test = y_scaled[:split], y_scaled[split:]
test_dates = daily_data.index[split:]

 #Define RBM + NN pipeline
rbm_nn = Pipeline(steps=[
    ('rbm', BernoulliRBM(n_components=64, learning_rate=0.06,
       n_iter=10, random_state=42)),
    ('mlp', MLPRegressor(hidden_layer_sizes=(128, 64), activation=
       'relu', max_iter=500, random_state=42))
])

 #Train the model
rbm_nn.fit(X_train, y_train)

 #Predict
y_pred_scaled = rbm_nn.predict(X_test)
y_pred = scaler_y.inverse_transform(y_pred_scaled)
y_true = scaler_y.inverse_transform(y_test)

rmse_demand, mae_demand, mape_demand = evaluate_forecast(y_true[:,
    0], y_pred[:, 0])
rmse_co2, mae_co2, mape_co2 = evaluate_forecast(y_true[:, 1],
   y_pred[:, 1])

print(f"{display_name} - Demand RMSE: {rmse_demand:.2f}, MAE: {
   mae_demand:.2f}, MAPE: {mape_demand:.2f}%")
print(f"{display_name} -  C O   RMSE: {rmse_co2:.2f}, MAE: {mae_co2
   :.2f}, MAPE: {mape_co2:.2f}%")
```

```
    plt.figure(figsize=(14, 5))
    plt.plot(test_dates, y_true[:, 0], label='Actual Demand')
    plt.plot(test_dates, y_pred[:, 0], label='Predicted Demand (RBM +
        NN)')
    plt.title(f"{display_name}     Electricity Demand Forecast (RBM +
        NN)")
    plt.legend()
    plt.grid(True)
    plt.show()

    plt.figure(figsize=(14, 5))
    plt.plot(test_dates, y_true[:, 1], label='Actual  C O  Emissions')
    plt.plot(test_dates, y_pred[:, 1], label='Predicted  C O  (RBM +
        NN)')
    plt.title(f"{display_name}      C O  Emissions Forecast (RBM + NN)
        ")
    plt.legend()
    plt.grid(True)
    plt.show()

     #Export Results to CSV
    result_df = pd.DataFrame({
        'Date': test_dates,
        'Region': display_name,
        'Actual_Demand': y_true[:, 0],
        'Predicted_Demand': y_pred[:, 0],
        'Actual_CO2': y_true[:, 1],
        'Predicted_CO2': y_pred[:, 1]
    })
    result_df.to_csv(f"rbm_nn_forecast_{region_code.lower()}.csv",
        index=False)

     Return metrics
    return {
        'Region': display_name,
        'Demand': {'RMSE': rmse_demand, 'MAE': mae_demand, 'MAPE':
            mape_demand},
        'CO2': {'RMSE': rmse_co2, 'MAE': mae_co2, 'MAPE': mape_co2}
    }

     Run for both California and Texas
cal_metrics = rbm_forecasting("CAL", "California")
tex_metrics = rbm_forecasting("TEX", "Texas")
```