NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Andreas Gravrok

Electricity consumption forecasting with Transformer models

An analysis of performance, generalizability and explainability

Master's thesis in Computer Science
Supervisor: Kshitij Sharma
June 2023

Andreas Gravrok

Electricity consumption forecasting with Transformer models

An analysis of performance, generalizability and explainability

Master's thesis in Computer Science
Supervisor: Kshitij Sharma
June 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Abstract

This study presents a comprehensive evaluation of various machine learning and deep learning models, with a particular focus on Transformer models, for short-term electricity consumption forecasting. The models evaluated include Ridge Regression, Gradient Boosting Model (GBM), Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), and Transformer models. The performance, generalizability, and explainability of these models were assessed using multiple datasets and under various conditions.

The Transformer model consistently outperformed LSTM and BiLSTM models across various datasets and lookback horizons, demonstrating its potential as a powerful tool for energy consumption forecasting. Interestingly, the Ridge Regression model also showed robust performance on certain datasets, challenging the common assumption that more complex models necessarily yield better performance.

The primary contribution of this study is the application and evaluation of Transformer models in the domain of Short Term Load Forecasting. The findings offer preliminary evidence that these models, which are traditionally used in Natural Language Processing tasks, may also have some potential in this domain. This study represents an initial exploration, and it is hoped that it will encourage further research into the applicability of Transformer models in energy consumption forecasting.

Despite the valuable insights, the study also acknowledges several limitations, including the specific conditions under which the models were evaluated and the limited range of models considered. These limitations highlight the need for further research to explore the performance of other models under diverse conditions, and to delve deeper into the concept of explainability in machine learning.

Sammendrag

Denne studien presenterer en omfattende evaluering av forskjellige maskinlærings- og dyp læringsmodeller, med et spesielt fokus på Transformer-modeller, for korttidsprognoser for strømforbruk. Modellene som ble evaluert inkluderer Ridge Regression, Gradient Boosting Model (GBM), Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), og Transformer-modeller. Ytelsen, generaliserbarheten og forklarbarheten til disse modellene ble vurdert ved hjelp av flere datasett og under forskjellige forhold.

Transformer-modellen presterte konsekvent bedre enn LSTM og BiLSTM-modellene på tvers av forskjellige datasett og tilbakeblikkshorisonter, noe som demonstrerer dens potensiale som et kraftig verktøy for energiforbruksprognoser. Interessant nok viste Ridge Regression-modellen også robust ytelse på visse datasett, noe som utfordrer den vanlige antagelsen om at mer komplekse modeller nødvendigvis gir bedre ytelse.

Hovedbidraget fra denne studien er implementeringen og evalueringen av Transformer-modeller i domenet for kortsiktig belastningsprognose. Funnene gir foreløpig bevis for at disse modellene, som tradisjonelt brukes på problemer med naturlig språkbehandling, også kan ha noe potensial i dette domenet. Denne studien representerer en initial utforskning, og det håpes at det vil oppmuntre til ytterligere forskning på anvendbarheten av Transformer-modeller i energiforbruksprognoser.

Til tross for de verdifulle innsiktene, anerkjenner studien også flere begrensninger, inkludert de spesifikke forholdene under hvilke modellene ble evaluert og det begrensede utvalget av modeller som ble vurdert. Disse begrensningene fremhever behovet for ytterligere forskning for å utforske ytelsen til andre modeller under diverse forhold, og for å gå dypere inn i konseptet med forklarbarhet i maskinlæring.

Contents

1 Introduction and Motivation **6**
- 1.1 Background and Context . . . 6
- 1.2 Research Objective and Questions . . . 6
- 1.3 Research Methodology . . . 6
- 1.4 Summary of Key Findings . . . 7
- 1.5 Thesis Structure . . . 7

2 Related Work **7**
- 2.1 Introduction . . . 7
- 2.2 Literature Search Strategy . . . 8
- 2.3 Detailed Analysis . . . 9
  - 2.3.1 Theme 1: Performance Validation of Modern Recurrent Neural Network Structures (LSTM, GRU) . . . 9
  - 2.3.2 Theme 2: Hyperparameter Tuning . . . 9
  - 2.3.3 Theme 3: Integration of Convolutional Neural Network (CNN) and LSTM . . . 9
  - 2.3.4 Theme 4: Ensembling . . . 9
  - 2.3.5 Theme 5: Accelerating Model Training . . . 10
  - 2.3.6 Theme 6: Novel Methods Benchmarked Against LSTM/GRU . . . 10
  - 2.3.7 Individual Paper Analyses . . . 10
- 2.4 Limitations of the State-Of-The-Art . . . 14
  - 2.4.1 Training Speed . . . 14
  - 2.4.2 Transferability . . . 14
  - 2.4.3 Explainability . . . 14

3 Methodology **15**
- 3.1 Granger Causality . . . 15
  - 3.1.1 The Granger Causality Test . . . 16
  - 3.1.2 Properties and Advantages of Granger Causality . . . 16
  - 3.1.3 Sliding Window Granger Causality . . . 17
- 3.2 Incorporating Temperature and Lagged Temperature Features . . . 18
  - 3.2.1 Temperature as a Feature . . . 18
  - 3.2.2 Lagged Temperature Features . . . 19
- 3.3 Incorporating Time Encodings as Features . . . 19
  - 3.3.1 Sinusoidal Time Encoding . . . 19
- 3.4 Incorporating Moving Mean as a Feature . . . 20
  - 3.4.1 Computing the Moving Mean . . . 20
  - 3.4.2 Benefits of Using Moving Mean as a Feature . . . 21
- 3.5 Incorporating Target without Cyclicality as a Feature . . . 21
  - 3.5.1 Computing the Target without Cyclicality . . . 22
  - 3.5.2 Benefits of Using Target without Cyclicality as a Feature . . . 22
- 3.6 Data Augmentation for Time Series . . . 22
  - 3.6.1 Generating Synthetic Historical Data . . . 23

3.6.2 Assessment of the Proposed Method 23
3.7 Performance on Production Settings 24
3.8 Generalizability 24
3.8.1 Pairwise-Generalizability Test 25
3.8.2 Leave-One-Out Generalizability 25
3.9 Explainability 26
3.9.1 Single Prediction Explainability (LIME) 26
3.9.2 Model Explainability 27
3.10 Ridge regression 27
3.10.1 Definition 27
3.10.2 Closed form solution 28
3.10.3 Stability and Robustness 28
3.10.4 Bias-Variance Trade-off 29
3.10.5 Regularization Path 29
3.10.6 Computational Efficiency 29
3.10.7 Conclusion 29
3.11 Gradient boosting models 29
3.11.1 Gradient Boosting 29
3.11.2 LightGBM: An Efficient Gradient Boosting Framework 30
3.12 Long Short-Term Memory (LSTM) Cells 31
3.12.1 Long Short-Term Memory (LSTM) Cells 32
3.13 Bidirectional Long Short-Term Memory (BiLSTM) 34
3.13.1 How BiLSTM Works 34
3.13.2 Differences Between BiLSTM and LSTM 34
3.13.3 Advantages of BiLSTM 34
3.14 Transformer Cells 35
3.14.1 Encoder 35
3.14.2 Decoder 36
3.14.3 Training and Inference 36
3.14.4 Key Advantages of Transformer Models 36
3.14.5 Activation Calculations in Transformer Cells 37
3.15 Datasets 38
3.15.1 Selected Datasets 38
3.15.2 Similarities in Datasets 40
3.15.3 Differences in Datasets 40
3.15.4 Omitted Datasets 41
3.16 Data pre-processing 43
3.16.1 Morocco Dataset 43
3.16.2 Paraguay Dataset 43
3.16.3 Spain Cities Dataset 43
3.16.4 Spain Houses Dataset 43
3.16.5 USA Dataset 43
3.16.6 Handling Missing Values 44
3.17 Training and Testing setup 44
3.17.1 48 hour lookback horizon 44

3.17.2 Longer lookback horizons . . . 46
3.17.3 Hyperparameter tuning . . . 47
3.17.4 Training and Testing Parameters . . . 48
3.17.5 Metrics . . . 48

4 Results **50**
4.1 Descriptive statistics . . . 50
4.2 Ridge . . . 51
4.3 GBM . . . 52
4.4 LSTM . . . 53
4.5 BiLSTM . . . 55
4.6 Transformer . . . 56
4.7 Comparing models on lookback horizons . . . 57
4.8 Production Settings . . . 59
4.9 Pairwise generalizability . . . 61
4.10 Leave-one-out generalizability . . . 63
4.11 Single Prediction Explainability . . . 63
4.12 model explainability . . . 64

5 Discussion **66**
5.1 Interpretation of results . . . 66
5.2 Implications . . . 68
5.3 Limitations . . . 69
5.4 Future Work . . . 70
5.5 Conclusion . . . 71

References **73**

1 Introduction and Motivation

1.1 Background and Context

As the world continues to advance in technological development and energy consumption continues to rise, there is a growing demand for accurate and reliable energy forecasting models. Ensuring a stable supply-demand balance in power systems, promoting energy efficiency, managing volatile energy markets, and reducing greenhouse gas emissions, all underscore the need for accurate short-term forecasting of electricity consumption [1].

Forecasting electricity consumption is a complex task due to the nonlinear and time-variant nature of electricity demand. Multiple factors including weather conditions, time of day, day of the week, holidays, and seasonal changes significantly influence electricity consumption patterns. This complexity and dynamism call for advanced machine learning models that can accurately capture and model these factors and their relationships.

1.2 Research Objective and Questions

The primary goal of this research is to explore the performance and applicability of Transformer-based models in the task of short-term electricity consumption forecasting. Transformer models, originally developed for Natural Language Processing (NLP) tasks, have shown remarkable performance due to their ability to handle long-range dependencies and parallel processing capabilities. The potential of Transformer models for time series forecasting has been relatively unexplored, and this research aims to fill that gap in the field of electricity demand forecasting.

The research questions guiding this study are as follows:

- How does the performance of Transformer models compare with other established machine learning models in short-term electricity consumption forecasting?
- How generalizable are Transformer models and other machine learning models across different datasets?
- To what extent can the predictions made by these models be explained to ensure transparency and trust in their predictions?

1.3 Research Methodology

The research methodology involved training and testing the Transformer model and comparator models (Ridge Regression, GBM, LSTM, and BiLSTM) on multiple datasets using various lookback horizons and additional features. Hyperparameters for all models were carefully tuned. The models' performance was evaluated based on their accuracy, computational efficiency, generalizability across different datasets, and explainability.

To unravel the 'black box' nature of these models, techniques like Local Interpretable Model-agnostic Explanations (LIME) and feature weight analysis to interpret and explain the models' predictions were applied. Understanding the factors that drive the predictions is crucial not only for building trust in the model's decisions but also for informing intervention strategies and policies.

1.4 Summary of Key Findings

The key findings of the study reveal that while the Transformer model outperformed LSTM and BiLSTM models across various datasets and lookback horizons, the Ridge Regression model demonstrated an unexpected robust performance on certain datasets. The computational efficiency of the Transformer model was significantly superior to LSTM and BiLSTM models due to its ability to leverage GPU-accelerated training. Furthermore, the Transformer model showed consistent improvement with the inclusion of additional features such as lagged temperature and time encodings.

Interestingly, despite their renowned capabilities, LSTM and BiLSTM models did not perform optimally in the single-step ahead forecasting task. This provides a contrasting perspective to the common narrative of the superiority of deep learning models in time-series forecasting tasks.

1.5 Thesis Structure

The thesis is structured as follows:

- Related Work: A comprehensive review of the current literature on machine learning models for electricity consumption forecasting.
- Methodology: A detailed explanation of the research design, the features and models used, the data collection, and analysis process.
- Results: Presentation and analysis of the research findings.
- Discussion: Interpretation of the results, discussion of their implications, comparison with existing literature, and identification of future research directions.

In conclusion, the findings from this research offer significant insights into the potential of Transformer models for electricity consumption forecasting and provide a foundation for future research in this area.

2 Related Work

2.1 Introduction

The intersection of deep learning and energy demand forecasting has become a burgeoning field of research due to the potential implications for sustainable energy management. With an increasing focus on the integration of renewable energy sources into the grid, accurate forecasting of energy demand has become paramount. This literature review aims to identify, analyze, and synthesize the existing body of knowledge on the application of recurrent neural networks (RNNs), particularly long short-term memory (LSTM) and gated recurrent unit (GRU) networks, in the context of weather-influenced energy demand forecasting.

2.2 Literature Search Strategy

The aim of the literature search strategy was to identify peer-reviewed articles emphasizing on the application of advanced deep learning techniques, including Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU), in energy demand forecasting, with a specific interest in the significant influence of weather, particularly temperature, on energy demand.

The initial search string for this exploration was: (”energy demand—consumption—load forecasting”) AND weather AND (LSTM—GRU—RNN), yielding 1890 results.

To refine the search and focus on the intended area of study, a number of keywords were strategically excluded, each serving to reduce the number of irrelevant papers:

- ’adversarial’ led to the exclusion of papers primarily focused on adversarial attacks and generation of realistic load data using GANs, resulting in 1680 papers.
- ’radiation’ and ’irradiation’ were excluded to avoid papers primarily focused on forecasting solar panel production, reducing the count to 1140.
- ’iot’ was eliminated to avoid papers focusing on predicting the energy usage of IoT devices, leaving 900 papers.
- ’energy storage’ was removed to omit papers focusing more on predicting battery performance, yielding 729 papers.
- ’ev’ was dropped to filter out papers more focused on battery performance for electric vehicles, leaving 703 papers.
- ’health’ was discarded to avoid papers focused on predicting battery health, resulting in 597 papers.
- ’room’ was omitted to exclude papers focused on predicting energy usage of items in a room, especially heaters, reducing the count to 533.
- ’driving’ was removed to filter out papers focusing on predicting energy usage for vehicles, yielding 510 papers.
- ’building energy’ was excluded to avoid papers that focus more on predicting energy consumption for individual buildings, which was considered as data with too small sample sizes and too large variances, resulting in 292 papers.

The final search string was thus: (”energy demand—consumption—load forecasting”) AND weather AND (LSTM—GRU—RNN) -adversarial -radiation -irradiation -iot -”energy storage” -ev -health -room -driving -”building energy”, yielding 292 results.

Out of these, 25 papers from the top 50 search results were selected for review. However, it is important to note the potential drawback of using negative keywords, as it may inadvertently filter out papers that merely mention related works in other fields.

2.3 Detailed Analysis

This section presents an in-depth examination of the literature, focusing on six primary thematic areas that emerged from the articles studied. Each theme is characterized by specific methodological approaches, findings, and implications.

2.3.1 Theme 1: Performance Validation of Modern Recurrent Neural Network Structures (LSTM, GRU)

The first theme encompasses studies that benchmarked the performance of modern recurrent neural network structures - Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) - against traditional forecasting models such as ARIMA and SVR. A notable finding across these papers is that LSTM and GRU, even in their basic, un-tuned forms, often surpassed classical models in forecasting accuracy. However, a challenge in this area is the lack of public availability of the datasets used in these studies, which limits reproducibility and further validation [2, 3, 4, 5].

2.3.2 Theme 2: Hyperparameter Tuning

The second theme emphasizes the exploration of hyperparameter tuning within different network types. Several techniques for hyperparameter optimization were employed across these studies, including genetic algorithms for determining the number of layers, neurons per layer, and other parameters such as learning rate, along with Coronavirus optimization algorithm (CVOA), Particle swarm optimization, and grid search parameter testing [6, 7, 8, 9].

2.3.3 Theme 3: Integration of Convolutional Neural Network (CNN) and LSTM

The third theme is characterized by studies exploring the combination of Convolutional Neural Networks (CNN) and LSTM or GRU networks. Two distinct methodologies were evident within this theme. Some studies applied CNN and LSTM/GRU independently to the data and later integrated the network outputs using a shallow dense network, while others employed the CNN initially to extract features from the data, which were then processed through an LSTM/GRU network [10, 11, 12, 13].

2.3.4 Theme 4: Ensembling

The fourth theme revolves around ensembling, where predictions from LSTM type networks and traditional models were combined. This amalgamation generally led to a reduction in variance and consequently, enhanced RMSE scores. The types of ensembles observed were diverse, including combinations such as Exponential smoothing time series + Residual dilated LSTM, ARIMA + LSTM, Prophet + LSTM, and SVR + Random Forest + LSTM [14, 15, 16, 17].

2.3.5 Theme 5: Accelerating Model Training

The fifth theme is centered on efforts to accelerate model training. In this context, two papers leveraged an autoencoder on the data, allowing the LSTM network to learn from the latent space variables. This approach not only maintained high-performance levels but also achieved up to a 50 percent reduction in execution time for forward pass [18, 19].

2.3.6 Theme 6: Novel Methods Benchmarked Against LSTM/GRU

The final theme involves studies testing innovative methods benchmarked against LSTM/GRU. These methods include Entangled LSTM and Enhanced LSTM (extensions of the standard LSTM structure), Temporal Convolution Network (TCN) which has demonstrated superior performance over LSTM for NLP tasks with long memory, an Ordinary Differential Equation (ODE) based generative approach, and Dual-Stage Bayesian Sequence to Sequence Embeddings [20, 21, 22, 23, 24].

In addition to these thematic areas, two papers employed distinct approaches. One study benchmarked a SARIMAX model against LSTM, while another utilized Principal Component Analysis to engineer data for more efficient learning. These standalone methods provide additional insights into the broad spectrum of techniques deployed in the field of energy demand forecasting. [25, 26]

2.3.7 Individual Paper Analyses

Performance Validation of RNN Structures

The first theme revolves around the validation of Recurrent Neural Networks (RNN) like structures. Patel et al. [2] demonstrated that LSTM and RNN networks performed worse than the SARIMA model for day-ahead predictions, but outperformed it for week-ahead predictions. Data from New York and India were used for their experimentation. Kumar et al. [3], in contrast, used a 7-node spark cluster for concurrent model training. Their study concluded that GRU network performs slightly better than LSTM, which in turn outperforms the RNN network.

Gui et al. [4] used a single shallow layer LSTM network to predict Ljubljana's load data with meteorological data from 2011. The study revealed that the inclusion of weather data significantly reduced the MAPE on week-ahead predictions. On the other hand, Islam et al. [5] concluded that LSTM networks performed superiorly to SVM models in a short-term load forecasting problem on a private dataset.

In conclusion, while SARIMA models may still hold their own in specific scenarios, RNN-based architectures, particularly LSTM and GRU networks, have demonstrated superior forecasting performance in various situations. The inclusion of exogenous variables like weather data has been shown to boost the accuracy of these deep learning models. These studies underline the importance of selecting the right architecture and data inputs for specific forecasting tasks.

Hyperparameter Tuning

Studies under this theme explored various ways to fine-tune hyperparameters for different types of networks. Mamun et al. [6] used a genetic algorithm for determining optimal hyperparameter values of LSTM networks, such as time lag, window size, number of neurons, and batch size. Their study showed superior performance compared to a normal LSTM network on a dataset from the Australian Energy Market Operator.

Torres et al. [7] utilized a random search optimization algorithm and a coronavirus optimization algorithm to determine the best LSTM networks. This was done by optimizing the number of hidden layers, units for each hidden layer, learning rates, and dropout rates. Both optimization schemes outperformed more traditional tree-based machine learning models, with a slight edge given to random search.

Lum et al. [8] conducted a grid search approach for determining optimal hyperparameters for a Temporal Convolutional Network. These included the number of convolutional filters, kernel size, and number of dilations. Their results suggested that the model focusing only on historical consumption performed better than the model considering economic factors.

Gundu et al. [9] used Particle Swarm Optimization to determine the optimal number of layers and number of nodes in each layer of a GRU network. Their model significantly outperformed other forecasting models such as ARIMA when tested on a US electricity dataset.

Overall, these studies highlight the importance of hyperparameter tuning in enhancing the performance of deep learning models. Various optimization algorithms, such as genetic algorithms, random search, grid search, and Particle Swarm Optimization, have been employed to optimize different parameters, from window size and batch size to learning rates and dropout rates. These findings reinforce that the choice of hyperparameters can significantly influence model performance.

CNN + LSTM

This theme focused on studies that combined Convolutional Neural Networks (CNN) with LSTM networks. Shao et al. [10] proposed a multi-channels and scales CNN–LSTM (MC-SCNN–LSTM) model. The CNN and the LSTM parts independently performed feature extraction before merging their outputs. This model demonstrated superior performance in very short-term, short-term, medium-term, and long-term load forecasting when tested against baseline models with shallow and fewer layers.

Wu et al. [11] proposed a CNN-GRU hybrid architecture, where the GRU module extracted feature vectors of time sequence data, and the CNN module extracted feature vectors of other high-dimensional data. This approach showed improved performance over benchmark models for short-term load forecasting.

Kuang et al. [12] proposed a hybrid model that used CNN to extract high-dimensionality features from consumption and temperature data, which were then passed through a LSTM network. This hybrid approach was tested on seasonal data—winter, summer, and spring autumn—and compared against a basic LSTM model and ARIMA model. The hybrid model demonstrated superior performance for short-term load forecasting.

Eskandari et al. [13] also used a CNN to extract high-dimensionality features from consumption and temperature data before sending the results through LSTM and GRU networks. The resulting model outperformed 14 other types of models published in previous papers.

In summary, the combination of CNN and LSTM or GRU networks has demonstrated superior performance in various load forecasting scenarios. The studies emphasize the benefit of utilizing the unique strengths of different network architectures, such as feature extraction from high-dimensional data by CNNs and sequence learning by LSTMs or GRUs. These hybrid models present a promising avenue for further research.

Ensembling

Ensembling techniques were employed by researchers to improve the performance of their models. Dudek et al. [14] implemented an ensemble model that won the Kaggle M4 forecasting competition in 2018. They adapted the model for a midterm load forecasting task, and it outperformed all selected benchmarks.

Pooniwala et al. [15] used a SARIMAX model and an LSTM model as inputs to a final network, which combined those predictions with other features to make a final prediction. The ensembled model outperformed each of the individual models.

Bashir et al. [16] employed a Prophet model to capture the autoregressive components of the data, while an LSTM network was trained to predict the residuals accurately. The sum of the predictions from both models provided the final prediction, significantly outperforming the benchmark models.

Muzumdar et al. [17] combined the predictions of an LSTM network, a Random Forest regressor, and an SVR regressor using a weighted average to create a final predicted load. This final model significantly outperformed the benchmark.

Collectively, these studies underscore the potential of ensemble techniques in improving forecasting accuracy. By combining the strengths of different models, ensemble methods can offer superior performance compared to individual models. This finding suggests that ensemble techniques could play a crucial role in advancing electricity load forecasting.

Faster Model Training

The studies under this theme aimed to improve training speed while maintaining or enhancing model accuracy. Ke et al. [18] employed a Stacked autoencoder to represent the input data in a lower dimension, which was then used to train a GRU network alongside the targets. This method resulted in a slightly more accurate model and marginally faster training.

Tao et al. [19] proposed an Encoder-Decoder LSTM model for multi-step forecasting. The encoding part of the network processed timesteps up to a certain point, while the decoder was trained to decode a continuation. This model performed comparably to other top-scoring models while being marginally quicker to train and predict than the CNN-LSTM hybrid.

Taken together, these studies indicate that innovative techniques can improve training speed without compromising model accuracy. Approaches like dimensionality reduction using

Stacked autoencoders and efficient LSTM architectures like Encoder-Decoder models can lead to faster training times. These findings highlight the importance of exploring novel methods for enhancing the efficiency of deep learning models.

Novel Methods

Researchers in this theme explored innovative methods for electricity load forecasting. Sharma et al. [20] investigated the impact of causal information between weather conditions and energy demand on forecasting performance using deep learning networks. They used a Bidirectional LSTM network with an extra layer for propagating hidden states and tested this new method on three different datasets, demonstrating its superiority over a baseline LSTM model.

Guo et al. [21] proposed reshaping the input for LSTM models based on weekly and monthly periodicity. The combined models outperformed a single, standard LSTM model. Hoang Anh et al. [22] used a Temporal Convolutional Network with a stride–dilation mechanism, which performed similarly to other state-of-the-art models on several benchmarks while significantly reducing the number of parameters.

Tsianikas et al. [23] proposed a neural ordinary differential equation (ODE) based generative approach for modeling electricity load sequences. Although a GRU network outperformed the proposed ODE method, the latter was found to be more robust against timestamp inconsistency and missing values issues during the integration of multi-source data.

Cameron-Muller et al. [24] proposed a Dual-Stage Bayesian Sequence to Sequence model, which offered probabilistic prediction intervals in addition to outperforming other benchmarks on four datasets.

In conclusion, these studies showcase a wide array of novel methods for electricity load forecasting. From integrating causal information to reshaping input data based on periodicity, the research demonstrates the potential of innovative techniques in enhancing forecasting performance. Such explorations lay the foundation for future research in this field.

Papers without Theme

There were two papers that did not fit into the above themes but still offered valuable insights. Johannesen et al. [25] applied Principal Component Analysis to identify the most significant factors in the data. The study showed that this method increased the performance of RNN and GRU networks but worsened the performance of LSTM networks on a Sydney load dataset.

Grigoryan et al. [26] used a broad range of socio-economic, energy-related, and weather-related features to predict energy consumption with a forecasting horizon of one month. Using 144 data points, with one point per month from January 2008 to December 2019, the researchers trained a SARIMAX model, which outperformed an LSTM model.

Overall, the comprehensive review of these studies provides a valuable insight into various methodologies and strategies adopted by researchers in the field of electricity load forecasting.

The themes identified not only help in understanding the current state of research in this area but also offer a starting point for the development of future forecasting models.

In summary, these studies emphasize that even though they do not fit into specific themes, innovative methods like Principal Component Analysis and the inclusion of socio-economic features can significantly influence model performance. The findings suggest that a diverse range of techniques and data inputs can be beneficial in improving load forecasting.

2.4 Limitations of the State-Of-The-Art

2.4.1 Training Speed

A transformer is a deep learning model [27] characterized by its self-attention mechanism, which differentially weights the significance of each part of the input data. Originally introduced for machine translation tasks, transformers were designed to avoid recursion, enabling parallel computation and significantly increasing training time while reducing sudden performance drops due to long dependencies. This is achieved by discarding the Markov property, which is an essential aspect of LSTM networks that require previously computed hidden states for each step.

Zeyer et al. [28] conducted a comparative analysis between transformers and a hybrid CNN-LSTM-based network for a speech recognition task. The results indicated that both models exhibited similar performance, but transformers demonstrated substantially reduced training time and enhanced model stability. Given the apparent advantages of transformers over LSTM networks, particularly as the data amount, network sizes, and training iterations increase, this thesis will explore the use of transformers in energy forecasting.

2.4.2 Transferability

Deep learning models often necessitate extensive data for training due to their complex architectures, which can be a limiting factor in their deployment across various applications [29]. Hence, the investigation of model transferability across different global locations is of significant interest, particularly in fields where data collection can be challenging or expensive. The potential correlations among historical consumption data, weather patterns, and current consumption may enable the application of insights gleaned from one geographical location to another. This cross-application of knowledge is made possible through transfer learning, an area of research that has shown substantial promise. Utilizing a pre-trained model and fine-tuning its last layers on data from the target area can lead to significant improvements in performance [30]. This approach has seen notable success in image classification tasks, where pre-trained networks can be retrained on custom classification tasks with only a few hundred training images, as opposed to the tens of thousands typically required to train a similar network from scratch [31, 32].

2.4.3 Explainability

Machine learning models exhibit a diverse range of strengths and weaknesses. Some models may prioritize accuracy over explainability, while others may sacrifice accuracy for increased explainability [33]. The trade-off between these two factors is crucial when developing machine learning applications.

Explainability is essential for several reasons. First, it facilitates the identification of errors within models [34]. A lack of understanding regarding a model's predictions makes it challenging to ascertain their correctness. Explainability also enables the comprehension of the rationale behind a model's predictions, which can aid in the development of new models or the improvement of existing ones [35].

Second, explainability helps build trust in models [36]. Users who lack understanding of a model's predictions may be less inclined to trust it. By providing a means for users to comprehend the logic behind predictions, explainability can foster trust.

Lastly, explainability is vital for ethical considerations [37]. When a model's decisions significantly impact people's lives, it is imperative that those affected understand the basis for such decisions. Explainability thus ensures that machine learning applications are used ethically and responsibly.

As electricity grids worldwide transition between various forecasting tools, from ARIMA to LSTM and potentially transformers in the future, an inherent trade-off exists between the strengths of these models and their explainability [38]. In most countries, maintaining and controlling the electricity grid is of utmost importance, as failures can have severe consequences [39]. Therefore, it is crucial for those responsible for balancing the grid to have access to models that are accurate, explainable, and maintainable.

3 Methodology

This chapter presents a comprehensive methodology for forecasting electricity consumption, focusing on incorporating various features and models. We begin with Granger Causality and the incorporation of different features, such as temperature, time encodings, moving mean, and target without cyclicality. Data augmentation through generating synthetic historical data is also discussed, along with model generalizability and explainability. Several models, including Ridge regression, gradient boosting, LSTM, BiLSTM, and Transformer Cells, are explored, detailing their properties and advantages.

The methodology of dataset selection, pre-processing, and handling missing values is explained. The chapter concludes with a discussion on the training and testing setup, which covers lookback horizons, hyperparameter tuning, and the metrics used to evaluate the performance of the models.

3.1 Granger Causality

Granger Causality is a statistical hypothesis test that aims to determine whether one time series can help predict another time series. Named after the Nobel Prize-winning economist Clive Granger, the concept of Granger Causality has been widely used in economics, finance, neuroscience, and various other fields to investigate the causal relationships among time series data [40]. In this chapter, the foundations of Granger Causality, its properties, and its application as an input feature in machine learning models will be discussed [41].

3.1.1 The Granger Causality Test

Given two time series, X and Y, the Granger Causality test aims to determine whether past values of X contain information that can improve the prediction of the current value of Y. Formally, X is said to Granger-cause Y if the inclusion of lagged values of X in a model for predicting Y improves the model's accuracy.

Consider the following linear regression models:

$$Y_t = \alpha_0 + \sum_{i=1}^{p}(\alpha_i Y_{t-i}) + \epsilon_t \tag{1}$$

$$Y_t = \beta_0 + \sum_{i=1}^{p}(\beta_i Y_{t-i}) + \sum_{i=1}^{q}(\gamma_i X_{t-i}) + \eta_t \tag{2}$$

Model (1) is an autoregressive (AR) model of order p for Y_t, using only past values of Y. Model (2) is an autoregressive distributed lag (ADL) model, which includes past values of both Y and X up to orders p and q, respectively.

To test whether X Granger-causes Y, we compare the prediction errors from both models. If the inclusion of lagged values of X in model (2) significantly reduces the prediction errors compared to model (1), we can conclude that X Granger-causes Y. This can be assessed using an F-test, comparing the residual sum of squares (RSS) from both models:

$$F = \frac{(RSS_1 - RSS_2)/q}{RSS_2/(T - 2p - q)} \tag{3}$$

Here, T is the number of observations, and RSS_1 and RSS_2 are the residual sums of squares from models (1) and (2), respectively. If the F-statistic is significant at a chosen significance level, we reject the null hypothesis that X does not Granger-cause Y.

3.1.2 Properties and Advantages of Granger Causality

- **Directionality**: Granger Causality tests for the directional influence of one time series on another, providing insights into the potential causal relationships among the variables. This is especially useful in fields such as economics and finance, where understanding the direction of causality can help inform policy decisions and investment strategies.
- **Multivariate Conditional Causality**: Granger Causality has shown to be further extended to multivariate settings[42], allowing for the assessment of conditional causal

relationships among multiple time series while controlling for the effects of other variables. This helps to isolate the direct causal relationships between the variables of interest and reduce the impact of spurious associations due to confounding factors.

- **Limitations and Assumptions**: Granger Causality is based on several assumptions, including linearity, stationarity, and a sufficiently large sample size. These assumptions can limit its applicability in certain scenarios, such as when dealing with nonlinear relationships or non-stationary data. Moreover, Granger Causality is purely a statistical test and does not guarantee true causality. It is essential to complement the results of Granger Causality tests with domain-specific knowledge to make meaningful inferences about causal relationships.
- **Feature Selection**: Granger Causality can be used as a feature selection technique for machine learning models that deal with time series data. By identifying the variables that Granger-cause the target variable, we can reduce the dimensionality of the input feature space, potentially improving the model's performance and interpretability.
- **Model Interpretability**: Incorporating Granger Causality tests into machine learning models can enhance their interpretability by providing insights into the causal relationships among the input features and the target variable. This is particularly valuable when building models for decision-making, as understanding the causal mechanisms can lead to more effective interventions and policy recommendations.
- **Time Series Forecasting**: Granger Causality can be used to inform the design of time series forecasting models. By identifying the variables that Granger-cause the target variable, we can build more accurate and robust forecasting models that incorporate relevant information from multiple time series.

3.1.3 Sliding Window Granger Causality

In this study, the Granger causality will be used as a sliding window input feature to the models [41]. A sliding window of Granger causalities can be used as a feature in time series analysis and machine learning models to capture the changing causal relationships between variables over time. This technique is particularly useful when dealing with non-stationary data or when the relationships among variables are expected to evolve over time. By computing Granger causality within sliding windows, we can create a set of dynamic features that account for the time-varying nature of the causal dependencies.

The first part in applying sliding window Granger causality is defining the window size and overlap. We choose an appropriate window size, which determines the number of observations used for each Granger causality computation. Smaller window sizes will be more sensitive to local changes in causal relationships, while larger window sizes will provide a more stable but potentially less responsive estimate of causality. Additionally, determine the degree of overlap between consecutive windows to balance the trade-off between the temporal resolution and computational complexity.

The second part is to compute Granger Causality for each window. Slide the window across the time series data, and for each position of the window, compute the Granger causality between the variables of interest. This results in a sequence of Granger causality values corresponding to different time intervals in the data, capturing the evolving causal dependencies over time.

The sliding window Granger causality can be computed using the following steps:

1. Define the window size W and overlap O.
2. Initialize the window position at the beginning of the time series data.
3. For each window position:
   (a) Extract the data within the window for the time series variables of interest.
   (b) Compute the Granger causality between the variables using equations (1) - (3) within the window.
   (c) Store the Granger causality value in the sequence of Granger causality values.
   (d) Move the window position forward by $(W - O)$ steps.
4. Continue until the window reaches the end of the time series data.

By using the sliding window Granger causality values as input features, we can incorporate the dynamic and evolving causal relationships between time series variables into our machine learning models. This technique is particularly valuable when dealing with non-stationary data or systems where the underlying dependencies among variables change over time. By calculating Granger causality within sliding windows, we can construct an adaptive feature that effectively represents the time-varying nature of causal relationships, potentially enhancing the performance and generalizability of our models.

3.2 Incorporating Temperature and Lagged Temperature Features

In this chapter, the incorporation of temperature and multiple lagged steps of temperature as features in machine learning models is discussed. The use of these features aims to enhance the performance and generalizability of the models by capturing the potential nonlinear and time-varying relationships between temperature, electricity consumption, and their historical values.

3.2.1 Temperature as a Feature

Temperature is an essential factor influencing electricity consumption, as it affects various aspects of energy demand such as heating, cooling, and other temperature-dependent processes [43]. By including temperature as an input feature, the direct and indirect effects of temperature on electricity consumption can be accounted for, potentially improving the accuracy and robustness of the models [44]. Moreover, the incorporation of temperature as a feature can help in capturing seasonal patterns in electricity consumption and understanding the underlying causal mechanisms driving energy demand [45].

3.2.2 Lagged Temperature Features

In addition to using the current temperature as an input feature, incorporating lagged values of temperature can help capture the temporal dependencies and dynamic relationships between temperature and consumption over time. These lagged features provide valuable information about the historical patterns and trends in temperature, which can improve the predictive power of the models by enabling them to learn from past data [46].

The inclusion of multiple lagged steps of temperature can also help account for potential nonlinear relationships between temperature and consumption. For instance, electricity consumption may not respond linearly to temperature changes, with different consumption behaviors at low, moderate, and high temperatures. By using multiple lagged steps, these nonlinearities can be captured, and the complex relationships between temperature, consumption, and their historical values can be better modeled.

3.3 Incorporating Time Encodings as Features

This chapter discusses the use of time encodings as features in machine learning models, specifically, the sinusoidal transformation of time-based variables. Sinusoidal time encoding can effectively capture the cyclical and seasonal patterns inherent in time-related variables, such as the hour of the day, day of the week, and day of the year. By incorporating these cyclical features into the models, it is possible to enhance their performance, generalization capabilities, and interpretability, especially in time series forecasting and analysis tasks.

3.3.1 Sinusoidal Time Encoding

The sinusoidal time encoding technique involves the transformation of time-related variables into sine and cosine components. This transformation takes advantage of the periodic nature of these variables, ensuring that they maintain a continuous cyclical representation. The following formulae are used for the sine and cosine transformations:

$$S(x) = sin(\frac{2\pi x}{N})$$

$$C(x) = cos(\frac{2\pi x}{N})$$

Here, $S(x)$ and $C(x)$ represent the sine and cosine components of the time-related variable x, and N is the maximum value of the variable, representing its periodicity.

The sinusoidal transformation is applied to three time-related variables: hour of the day, day of the week, and day of the year. The following transformations are performed on these variables:

For the hour of the day variable, $N = 24$, as there are 24 hours in a day. The sine and cosine components are computed as follows:

$$\text{hour\_of\_day\_sin} = \sin\left(\frac{2\pi \cdot \text{hour}}{24}\right) \tag{4}$$

$$\text{hour\_of\_day\_cos} = \cos\left(\frac{2\pi \cdot \text{hour}}{24}\right) \tag{5}$$

For the day of the week variable, $N = 7$, as there are 7 days in a week. The sine and cosine components are computed as follows:

$$\text{day\_of\_week\_sin} = \sin\left(\frac{2\pi \cdot \text{dayofweek}}{7}\right) \tag{6}$$

$$\text{day\_of\_week\_cos} = \cos\left(\frac{2\pi \cdot \text{dayofweek}}{7}\right) \tag{7}$$

For the day of the year variable, $N = 365$, as there are 365 days in a year (ignoring leap years). The sine and cosine components are computed as follows:

$$\text{day\_of\_year\_sin} = \sin\left(\frac{2\pi \cdot \text{dayofyear}}{365}\right) \tag{8}$$

$$\text{day\_of\_year\_cos} = \cos\left(\frac{2\pi \cdot \text{dayofyear}}{365}\right) \tag{9}$$

These formulas transform the original time variables (hour, day of the week, and day of the year) into continuous values ranging from -1 to 1, preserving the cyclical nature of these variables.

3.4 Incorporating Moving Mean as a Feature

This chapter focuses on the addition of the moving mean as a feature in machine learning models. Moving mean, also known as the rolling or moving average, is a widely-used technique for smoothing time series data to reduce noise and highlight underlying trends. When incorporated as a feature, the moving mean can provide valuable information about the local context of a time series, assisting the model in capturing patterns and dependencies across different time scales.

3.4.1 Computing the Moving Mean

The moving mean is calculated as the average of a fixed number of consecutive data points within a specified window, which slides over the time series data. The size of the window, or the rolling window, determines the level of smoothing applied to the data. A larger window size will result in a smoother representation of the time series, while a smaller window size will retain more details and fluctuations in the data. The following formula computes the moving mean:

$$MM_t = \frac{1}{W} \sum_{i=0}^{W-1} x_{t-i}$$

Here, MM_t represents the moving mean at time t, x_{t-i} denotes the data point at time $(t-i)$, and W is the size of the rolling window.

To calculate the moving mean of a 'Consumption' variable in a DataFrame using a specified rolling window size, the rolling(window) method is applied to the 'Consumption' column. This method returns a Rolling object that can be used to perform various operations on the data within the window, such as computing the mean. The mean is then computed for each window using the mean() method.

To prevent data leakage and ensure that the moving mean feature at time t only contains information from past data points, the computed moving mean is shifted forward by one time step using the shift(1) method.

3.4.2 Benefits of Using Moving Mean as a Feature

Incorporating the moving mean as a feature in machine learning models can provide several advantages:

- Noise Reduction: The moving mean smooths out the time series data, reducing noise and highlighting underlying trends. This can help the model focus on the essential patterns in the data and avoid being influenced by random fluctuations.
- Local Context: The moving mean provides information about the local context of the time series, which can be particularly useful for capturing dependencies and patterns across different time scales.
- Robustness: By using the moving mean as a feature, the model can become more robust to outliers and sudden changes in the data, as the moving mean is less sensitive to individual data points compared to raw values.

3.5 Incorporating Target without Cyclicality as a Feature

This chapter discusses the addition of the target without cyclicality as a feature in machine learning models. The target without cyclicality is computed by subtracting the moving mean from the original target variable (in this case, 'Consumption'). This feature aims to remove cyclical patterns from the target variable, focusing on the fluctuations that deviate from the local average. By including this feature, models can learn to better capture the irregularities and variations in the time series data.

3.5.1 Computing the Target without Cyclicality

To calculate the target without cyclicality, the moving mean of the target variable is first computed using a specified rolling window size, as described in the previous chapter. Then, the moving mean is subtracted from the original target variable at each time step. The following formula computes the target without cyclicality:

$$TWC_t = x_t - MM_t$$

Here, TWC_t represents the target without cyclicality at time t, x_t denotes the target variable at time t, and MM_t is the moving mean at time t.

Similar to the moving mean feature, to prevent data leakage and ensure that the target without cyclicality feature at time t only contains information from past data points, the computed feature is shifted forward by one time step using the shift(1) method.

3.5.2 Benefits of Using Target without Cyclicality as a Feature

Incorporating the target without cyclicality as a feature in machine learning models offers several advantages:

- Focus on Irregularities: By removing the cyclicality from the target variable, the models can focus on learning the irregularities and variations in the time series data, potentially improving their ability to capture the underlying dynamics.
- Robustness: The target without cyclicality feature can make models more robust to changes in the cyclical patterns of the data. Since the feature focuses on the deviations from the local average, the models are less sensitive to the specific cyclicality of the time series.
- Enhanced Feature Set: Including the target without cyclicality as a feature can provide additional information to the models, complementing other features such as the moving mean and time encodings. This can lead to a more comprehensive representation of the time series data, potentially improving the model's performance.

3.6 Data Augmentation for Time Series

Data augmentation is a technique commonly used in machine learning to generate additional training data by applying transformations to the original data. However, applying data augmentation techniques to time series data can be challenging due to the inherent temporal dependencies and constraints.

This chapter discusses a specific method for augmenting time series data by creating synthetic historical data. The method involves generating additional data points by shifting the original time series dataset backward in time and adding random noise to the target variable (Consumption) and the Temperature feature. However, it is important to note that this approach may still lead to data leakage and should be considered as an indication of how the model's performance could potentially improve with more data, rather than an actual improvement in the model's generalization capabilities.

3.6.1 Generating Synthetic Historical Data

The proposed method generates synthetic historical data by creating multiple copies of the original dataset and shifting them backward in time. In each iteration, the 'DateTime' column is modified by subtracting a multiple of the total time span of the dataset. The Consumption and Temperature columns are then perturbed by adding random noise proportional to their original values, with a specified noise level. Finally, the augmented datasets are concatenated with the original dataset in chronological order.

This method preserves the temporal dependencies in the original data while creating variations in the target variable and features. By generating synthetic historical data, the model can learn from a larger and more diverse set of training examples, potentially improving its ability to generalize to unseen data.

However, it is crucial to recognize the potential for data leakage when employing this method. Data leakage refers to a situation where information from outside the training dataset is used to create the model. This can result in overly optimistic performance estimates during model validation, as the model has had access to data it shouldn't have during training.

In the context of our synthetic data generation method, just shifting the data and adding 2% noise may still lead to data leakage because the synthetic data points are highly correlated with the original data points. For example, if we simply shift the data by one day and add noise, a synthetic Monday's data point will be very similar to the original Tuesday's data point. If the original Tuesday's data point is in the validation set and the synthetic Monday's data point is in the training set, the model may perform well on the validation set, not because it has generalized well, but because it has effectively "seen" similar data during training.

This means that the performance improvements observed in the model when trained on the augmented dataset may not be entirely due to the model's ability to generalize better, but rather due to the increased similarity between the training and validation data. This can mask the model's true predictive capability and provide a false sense of confidence in its performance. Therefore, it's important to be aware of this risk and consider additional techniques to ensure the validity of the model's performance, such as rigorous cross-validation or the use of completely independent datasets for validation and testing.

3.6.2 Assessment of the Proposed Method

Despite the risk of data leakage, the method of generating synthetic historical data for time series augmentation can provide insights into the potential performance improvements that could be achieved with more data. By assessing the model's performance on the augmented dataset, one can get an indication of how well the model might perform if additional genuine historical data were available.

However, it is essential to treat the performance results obtained using this method with caution and not to rely solely on them to evaluate the model's true generalization capabilities. To obtain a more accurate assessment of the model's performance, it is necessary to test it on completely independent and unseen data that is not affected by data leakage.

In conclusion, the proposed method can be a valuable tool for understanding the potential impact of additional data on the model's performance, provided that its limitations and the risk of data leakage are properly considered.

3.7 Performance on Production Settings

This subsection presents the evaluation of model performance in various production settings, which reflect real-world scenarios where the first few hours of data might be unavailable. The unavailability of data can arise due to several reasons, such as the need to adjust the model based on inflows and outflows in order to estimate the actual consumption. When a model is trained on real consumption, it is essential to evaluate its performance on the same data to obtain accurate and reliable predictions.

The performance of the models was assessed under different production settings, where the first x hours of data were unavailable. The production settings evaluated were 2, 6, 24, and 48 hours. For each setting, the models were trained and tested on the adjusted dataset with the specified number of unavailable hours, and their performance was compared.

The models evaluated include the Ridge Regression model, Gradient Boosting Model (GBM), LSTM model, BiLSTM model, and Transformer model. For each production setting, the models were trained and evaluated on a range of datasets that represent different electricity consumption scenarios.

The evaluation of the models under these production settings provides valuable insights into their robustness and adaptability in real-world situations. It enables the identification of models that can perform well even with limited data availability, which is crucial for practical applications. By analyzing the performance of the models in various production settings, it is possible to determine their suitability for deployment in real-world energy consumption prediction tasks, ensuring that the models can provide accurate and reliable forecasts even when faced with data constraints.

3.8 Generalizability

The generalizability of a machine learning model is an essential aspect of its performance, as it indicates the model's ability to perform well on new, unseen data. For energy consumption forecasting, it is crucial that the models can generalize across different regions and scenarios, as the demand patterns and conditions can vary significantly. This section presents the evaluation of the generalizability of the models, including the Ridge Regression model, Gradient Boosting Model (GBM), LSTM model, BiLSTM model, and Transformer model.

3.8.1 Pairwise-Generalizability Test

To assess the generalizability of the models, a pairwise-generalizability test was conducted. In this test, each model was trained on one dataset and then evaluated on all other datasets, representing different regions and scenarios. This approach provides a comprehensive understanding of the model's ability to generalize across various conditions, highlighting its potential for deployment in real-world energy consumption prediction tasks.

Furthermore, if the models exhibit good generalizability, they can potentially be used for transfer learning to other regions. Deep learning models often require a significant amount of training data to capture the underlying patterns and relationships. By leveraging the pre-trained models with good generalization performance, it may be possible to achieve satisfactory results with less training data for new regions.

The generalizability test was conducted using the following procedure for each model:

- Load the trained model for a specific dataset, such as Morocco, Paraguay, Spain (cities average), Spain (houses), or the USA.
- For each of the other datasets, preprocess the data by adding lagged timesteps and scaling the features and target variables using a StandardScaler.
- Split the preprocessed data into training and testing sets, maintaining the temporal order of the data.
- Evaluate the model's performance on the test set by computing the Mean Absolute Percentage Error (MAPE).

The pairwise-generalizability test results provide insights into the models' capability to perform well on unseen data from different regions and scenarios. By analyzing the test results, it is possible to identify the models that exhibit strong generalization performance, which is crucial for ensuring accurate and reliable energy consumption forecasts in real-world applications and facilitating transfer learning to other regions with limited data.

3.8.2 Leave-One-Out Generalizability

Another approach to assess the generalizability of the models is the leave-one-out generalizability test. In this test, one dataset is held out as the test set, while the remaining datasets are combined and used as the training set. This process is repeated for each dataset, allowing the evaluation of the model's performance on unseen data from different regions. The leave-one-out generalizability test primarily evaluates the model's bias, as it assesses the ability of the model to generalize across different datasets, capturing the underlying patterns despite differences in the data.

On the other hand, the pairwise generalizability test assesses the model's variance, as it evaluates the model's performance when trained on one dataset and tested on another. This test determines whether a model overfits the training data or if it can adapt to the variations present in different datasets.

The leave-one-out generalizability test was conducted for all models using the following procedure:

- For each dataset, hold it out as the test set and use the remaining datasets as the training set.
- Preprocess the training and test datasets
- Concatenate the preprocessed training datasets to form a single training set.
- Fit the respective model on the combined training set.
- Evaluate the model on the test set using performance metrics such as Mean Absolute Percentage Error (MAPE)

This process was repeated for each of the considered models, providing insights into their ability to generalize to new, unseen data from different regions, as well as their bias and variance characteristics.

3.9 Explainability

The concept of explainability, or interpretability, in machine learning models is of increasing importance, particularly in areas where decision justification is required. It involves understanding the underlying mechanics of complex models and how they make decisions. While advanced machine learning models often deliver superior performance, this often comes at the cost of interpretability, leading to models that behave as 'black boxes'. In this research, two facets of explainability were explored: model explainability and single prediction explainability.

3.9.1 Single Prediction Explainability (LIME)

To explain individual predictions of the models, the Local Interpretable Model-agnostic Explanations (LIME) method was employed. LIME is an approach that explains the predictions of any classifier in a faithful way, by approximating it locally with an interpretable model.

The general idea of LIME is to generate a new dataset consisting of perturbed samples, get the predictions for these instances, and then weigh them by the proximity of these instances to the instance of interest. A simple model is learned on this dataset and used to explain the instance of interest.

However, it is crucial to note that LIME has limitations, particularly with sequence data, which means it is not applicable to all machine learning models. Specifically, LIME does not handle sequence data, which is a type of data that LSTM and BiLSTM models deal with. Therefore, the LIME method could not be used to interpret individual predictions of LSTM and BiLSTM models in this research. This limitation underscores the ongoing challenges in the field of explainability in machine learning and the need for more versatile explainability techniques that can accommodate various types of data and models.

3.9.2 Model Explainability

Model explainability aims to elucidate the underlying decision-making process of a machine learning model. It involves interpreting how the model forms predictions from the given inputs. A crucial element of this interpretability is the concept of feature importance, which provides insights into the relative impact of each input feature on the model's predictions.

When discussing feature importance, particularly for simpler models such as Ridge Regression and Gradient Boosting Models, the concept of "heavily weighted features" often comes into play. These are the features or variables that have the most substantial influence on the model's predictions. For instance, in Ridge Regression, the coefficients of the regression model signify the importance of each feature. A larger absolute value of a coefficient suggests a more important feature, meaning it has a stronger influence on the prediction.

For a Gradient Boosting Model, feature importance is calculated based on how frequently a feature is utilized to split the data across all decision trees, coupled with the improvement in the model's performance as a result of these splits.

Understanding the most heavily weighted features can offer crucial insights into the data and the model's decision-making process. It helps determine which features significantly influence the predictions and allows the identification of any potential data biases.

In the context of more complex models, such as deep learning architectures, computing feature importance can be more intricate. Prior to the release of TensorFlow v2, the SHapley Additive exPlanations (SHAP) library was used to calculate a unified measure of feature importance. SHAP assigns the contribution of each feature to each prediction in a fair way. However, the use of the SHAP library with TensorFlow v2 models is no longer recommended, as it disables essential features specific to TensorFlow 2.x, including eager execution.

When interpreting the importance of features, especially for the top 5 heavily weighted ones, a positive weight indicates a feature that increases the output prediction's value. In contrast, a weight of zero implies that the feature has no informational gain to the output prediction's value. The magnitude of the weight signifies the strength of the influence: the larger the magnitude, the stronger the effect.

3.10 Ridge regression

Ridge Regression[47], also known as Tikhonov regularization or L2 regularization, is a regularization technique employed to enhance the generalization performance of linear regression models. It addresses the problem of multicollinearity and overfitting by adding a penalty term to the ordinary least squares (OLS) objective function.

3.10.1 Definition

Given a dataset $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ with N observations and p input features, the objective of linear regression is to find a weight vector w and a bias term b that minimize the residual sum of squares (RSS):

$$RSS(w,b) = \sum_{i=1}^{N}(y_i - (w^T x_i + b))^2$$

In Ridge Regression, a penalty term proportional to the squared L2-norm of the weight vector is added to the objective function, resulting in the following optimization problem:

$$J(w,b) = RSS(w,b) + \lambda\|w\|^2$$

Here, $\lambda \geq 0$ is the regularization parameter that controls the amount of shrinkage applied to the weight vector. The regularization term, $\lambda\|w\|^2$, discourages large weights and encourages a more stable and robust solution.

When $\lambda = 0$, Ridge Regression reduces to the standard OLS linear regression. As λ increases, the penalty term becomes more dominant, leading to a greater shrinkage of the weights. The optimal value of λ can be determined using techniques such as cross-validation.

3.10.2 Closed form solution

To find the optimal weight vector w and bias term b that minimize $J(w,b)$, we can differentiate the objective function with respect to w and b, and set the resulting gradients to zero:

$$\nabla_w J(w,b) = -2X^T(y - Xw - b1) + 2\lambda w = 0$$
$$\nabla_b J(w,b) = -2(y - Xw - b1)^T 1 = 0$$

Here, X is the $N \times p$ feature matrix, y is the $N \times 1$ target vector, and 1 is an $N \times 1$ vector of ones.

Solving these equations for w and b yields the closed-form solution for Ridge Regression:

$$w = (X^T X + \lambda I)^{-1} X^T (y - b1)$$
$$b = \frac{1}{N}(y - Xw)^T 1$$

3.10.3 Stability and Robustness

Ridge Regression mitigates the issue of multicollinearity among input features, providing a more stable and robust solution compared to OLS linear regression[48]. By shrinking the weights, Ridge Regression reduces the variance of the model and improves its generalization performance on unseen data.

3.10.4 Bias-Variance Trade-off

Ridge Regression balances the trade-off between bias and variance by controlling the complexity of the model through the regularization parameter λ[48]. This leads to better out-of-sample predictions, especially when the underlying relationship between the input features and the target variable is not strictly linear. Ridge Regression's L2 regularization also has advantages over L1 regularization, such as rotational invariance[49].

3.10.5 Regularization Path

Ridge Regression provides a continuous regularization path, allowing for the exploration of different model complexities by varying the regularization parameter λ. This can help in understanding the importance of input features and selecting the optimal level of regularization.

3.10.6 Computational Efficiency

Ridge Regression has a closed-form solution[48], which can be computed efficiently using matrix factorization techniques, such as the Cholesky decomposition or singular value decomposition (SVD). This makes it suitable for large-scale problems with many input features and observations.

3.10.7 Conclusion

In conclusion, Ridge Regression is an effective regularization technique for linear regression models that enhances their generalization performance, stability, and robustness. By incorporating a penalty term based on the squared L2-norm of the weight vector, Ridge Regression addresses the issues of multicollinearity and overfitting, providing a more reliable solution compared to the standard OLS linear regression. The closed-form solution, efficient computational methods, and the ability to control the model complexity through the regularization parameter make Ridge Regression a valuable tool for tackling regression problems, particularly when the input features are correlated or the underlying relationship between the input features and the target variable is not strictly linear.

3.11 Gradient boosting models

Gradient Boosting Models (GBMs) are an ensemble learning technique used for addressing regression and classification problems. They have gained significant attention due to their ability to provide state-of-the-art results on a wide range of tasks, by combining multiple weak learners to form a strong predictive model. This section presents a comprehensive and academic discussion on the workings of gradient boosting models and concludes with an overview of the LightGBM framework.

3.11.1 Gradient Boosting

Gradient Boosting is a sequential learning technique that builds an ensemble of weak learners, typically decision trees, in a stage-wise manner[50]. The algorithm seeks to minimize the loss function of the overall model by adding new weak learners that correct the errors made by the previous learners. The intuition behind gradient boosting is to exploit the gradient information of the loss function to guide the training of subsequent weak learners, thereby optimizing the model's performance[50].

Formally, given a differentiable loss function $L(y, F(x))$, where y represents the true target values and $F(x)$ represents the ensemble prediction, the goal is to find the optimal function $F(x)$ that minimizes the expected value of the loss function. The optimization is performed by constructing an ensemble of weak learners, $h_m(x)$, with $m = 1, 2, \ldots, M$, and combining them in the form:

$$F_m(x) = F_0(x) + \sum_{m=1}^{M} \rho_m h_m(x)$$

Here, $F_0(x)$ is the initial prediction, $h_m(x)$ are the weak learners, and ρ_m are the corresponding learning rates.

The algorithm begins with an initial model, $F_0(x)$, which can be a simple constant or a weak learner that predicts the average of the target values. At each stage m, a new weak learner is fitted to the negative gradient of the loss function, also known as the pseudo-residuals:

$$r_{mi} = -\frac{\partial L(y_i, F_{m-1}(x_i))}{\partial F_{m-1}(x_i)}$$

The weak learner is then incorporated into the ensemble, and its contribution is determined by optimizing the learning rate, ρ_m, through line search:

$$\rho_m = \arg\min_\rho \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i + \rho h_m(x_i)))$$

This procedure is repeated for M stages, resulting in a final ensemble model, $F_M(x)$, that minimizes the overall loss function.

Hyperparameters such as the number of weak learners (M), the maximum depth of the decision trees, and the learning rate (ρ) play a crucial role in controlling the complexity of the model and preventing overfitting.

3.11.2 LightGBM: An Efficient Gradient Boosting Framework

LightGBM (Light Gradient Boosting Machine) is an open-source gradient boosting framework that employs tree-based learning algorithms[51]. Developed by Microsoft, it addresses some of the limitations of traditional gradient boosting models, such as high memory usage, long training times, and poor scalability to large datasets.

LightGBM introduces several innovations that result in a more efficient and accurate implementation of gradient boosting[51]:

- **Gradient-based One-Side Sampling (GOSS)**: LightGBM utilizes a novel sampling technique to select a subset of the training instances based on the magnitude of their gradients. Instances with larger gradients, which contribute more to the overall loss, are retained[51].
- **Exclusive Feature Bundling (EFB)**: In many real-world datasets, feature values are sparse, and many features are mutually exclusive, meaning that they are never non-zero simultaneously. LightGBM takes advantage of this property by bundling exclusive features together, effectively reducing the number of features and decreasing the memory consumption and training time[51].
- **Leaf-wise Tree Growth**: Unlike traditional depth-wise or level-wise tree growth strategies, LightGBM employs a leaf-wise tree growth approach[51]. In this method, the decision tree is grown by selecting the leaf with the maximum delta loss and splitting it, regardless of the tree depth. Although this can lead to deeper trees, it generally results in a more accurate model with similar complexity[51].
- **Categorical Feature Support**: LightGBM can handle categorical features directly by finding the optimal split points for categorical variables, without the need for manual one-hot encoding or other preprocessing techniques[51]. This can significantly reduce memory usage and improve computational efficiency.
- **Parallel and GPU Learning**: LightGBM supports parallel learning using multi-core CPUs, as well as GPU learning, enabling the framework to scale efficiently to large datasets and high-performance hardware[51].

In summary, LightGBM is an efficient and scalable gradient boosting framework that incorporates innovative techniques such as Gradient-based One-Side Sampling, Exclusive Feature Bundling, and leaf-wise tree growth, leading to faster training times and improved model accuracy. With its ability to handle large-scale datasets, support for categorical features, and parallel and GPU learning capabilities, LightGBM has become a popular choice for a wide range of regression and classification tasks.

3.12 Long Short-Term Memory (LSTM) Cells

To understand Long Short-Term Memory (LSTM) cells, we must first introduce Recurrent Neural Networks (RNNs). RNNs are a class of artificial neural networks designed for sequential data processing and modeling tasks, such as time series forecasting, natural language processing, and speech recognition[52]. Unlike feedforward neural networks, RNNs possess a unique architecture that allows them to maintain a hidden state that acts as a form of memory, enabling them to learn and remember patterns across sequences of data[53]. This is achieved by incorporating loops within the network, allowing information to persist and be reused as the network processes subsequent inputs.

However, RNNs suffer from two significant limitations: the vanishing gradient and exploding gradient problems[53]. In the vanishing gradient problem, gradients tend to become very small as they are backpropagated through the network during training, which can cause the network to learn slowly or even stop learning altogether. Conversely, the exploding gradient problem arises when gradients become too large, leading to unstable training and poor generalization. Both issues are particularly pronounced when dealing with long input sequences, as the influence of earlier inputs tends to diminish or become too dominant over time.

3.12.1 Long Short-Term Memory (LSTM) Cells

LSTM cells were introduced by Hochreiter and Schmidhuber in 1997 as a solution to the vanishing and exploding gradient problems in RNNs. LSTMs are a type of recurrent neural network architecture that specifically addresses these issues by incorporating a more sophisticated memory cell capable of learning and retaining long-range dependencies in the input data[54]. The LSTM cell achieves this through a combination of gating mechanisms that carefully regulate the flow of information within the cell[55].

An LSTM cell consists of three main components: the input gate, the forget gate, and the output gate, along with a memory cell state. These gates work in concert to determine how information should be stored, updated, or retrieved from the memory cell state.

- **Input gate**: The input gate is responsible for determining which information from the current input and previous hidden state should be allowed to enter the memory cell state. It does so by utilizing a sigmoid activation function, which outputs values between 0 and 1, signifying how much of each piece of information should be stored.
- **Forget gate**: The forget gate decides which information from the current memory cell state should be discarded or retained. Like the input gate, it uses a sigmoid activation function to produce values between 0 and 1, indicating how much of each memory content should be kept.
- **Output gate**: The output gate controls which information from the memory cell state should be used to produce the cell's output and the next hidden state. It also employs a sigmoid activation function to weigh the importance of the information, while a separate activation function, such as the hyperbolic tangent (tanh), is used to scale the output of the memory cell state.

Memory cell state: The memory cell state, often denoted as 'C', is the core of the LSTM cell. It is a continuous representation of the information stored in the cell, which can be updated by the input and forget gates and accessed by the output gate.

The interplay of these gates allows the LSTM cell to learn when to store, update, or discard information based on the input sequence's context. This selective gating mechanism enables LSTMs to capture long-range dependencies, effectively overcoming the vanishing and exploding gradient problems that plague traditional RNNs.

In summary, Long Short-Term Memory (LSTM) cells are a powerful and versatile improvement upon traditional Recurrent Neural Networks (RNNs), designed to address the vanishing and exploding gradient problems. By incorporating a sophisticated memory cell with gating mechanisms, LSTMs can effectively learn and retain long-range dependencies in sequential data.

Activation of LSTM Cells

Now, let's explore how the activation of each gate and the memory cell state in an LSTM cell is calculated. We will go through each gate and the memory cell state one by one, explaining their corresponding mathematical equations.

1. Input gate

The input gate consists of two parts: the input modulation gate (i_t) and the candidate memory cell state ($\tilde{C}_t$). The input modulation gate determines how much of the new input should be stored, while the candidate memory cell state represents a filtered version of the input. The equations for the input modulation gate and the candidate memory cell state are:

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi})$$
$$\tilde{C}_t = tanh(W_{i\tilde{C}}x_t + b_{i\tilde{C}} + W_{h\tilde{C}}h_{t-1} + b_{h\tilde{C}})$$

Here, x_t is the input vector at time step t, h_{t-1} is the hidden state at the previous time step, W and b denote weight matrices and bias vectors, and σ is the sigmoid activation function.

2. Forget gate

The forget gate (f_t) determines which information in the memory cell state should be retained or discarded. The equation for the forget gate is:

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf})$$

3. Memory cell state

The memory cell state (C_t) is updated by combining the input gate's output and the forget gate's output. The equation for the memory cell state is:

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

Here, $\odot$ represents element-wise multiplication.

4. Output gate

The output gate (o_t) controls the information flow from the memory cell state to the output and the next hidden state. The equation for the output gate is:

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho})$$

5. Hidden state

Finally, the hidden state (h_t) is computed using the output gate and the memory cell state. The equation for the hidden state is:

$$h_t = o_t \odot tanh(C_t)$$

3.13 Bidirectional Long Short-Term Memory (BiLSTM)

Bidirectional Long Short-Term Memory (BiLSTM) is an extension of the standard LSTM architecture, designed to better capture context from both past and future elements in a sequence[56]. While LSTM models process the input sequence in a unidirectional manner, either forward or backward, BiLSTM models process the sequence in both directions simultaneously[57]. This bidirectional processing provides a more comprehensive understanding of the input sequence, improving the model's ability to capture complex patterns and dependencies.

3.13.1 How BiLSTM Works

A BiLSTM consists of two separate LSTM layers, each with its own set of input, forget, and output gates, and memory cell states. One LSTM layer processes the input sequence in the forward direction, while the other processes it in the backward direction[57]. The outputs of both LSTM layers are combined at each time step, either through concatenation or another merging operation, to produce a single output for the BiLSTM model.

The forward LSTM layer starts at the beginning of the input sequence and processes it in the usual left-to-right order, while the backward LSTM layer starts at the end of the sequence and processes it in the opposite right-to-left order. By processing the sequence in both directions, the BiLSTM model effectively captures both past and future context for each element in the sequence.

3.13.2 Differences Between BiLSTM and LSTM

The primary difference between BiLSTM and standard LSTM models lies in their approach to processing input sequences. While LSTM models process the sequence in a unidirectional manner, BiLSTM models process the sequence bidirectionally, incorporating information from both past and future elements. This is achieved by employing two separate LSTM layers, one for each direction, which are combined to produce the final output.

Another difference between the two architectures is in the output representation. In LSTM models, the output at each time step is based solely on the hidden state produced by the LSTM cell. In contrast, the output of a BiLSTM model is derived from the hidden states of both the forward and backward LSTM layers. These hidden states are typically combined through concatenation or another merging operation, resulting in a richer output representation that captures context from both directions.

3.13.3 Advantages of BiLSTM

BiLSTM models offer several advantages over standard LSTM models, particularly when it comes to capturing context and dependencies in sequential data[57]:

- **Improved context capture**: By processing the input sequence in both forward and backward directions, BiLSTM models can capture context from both past and future elements, leading to a more comprehensive understanding of the sequence.
- **Better performance on complex tasks**: BiLSTM models often outperform standard LSTM models on tasks that require the understanding of long-range dependencies or bidirectional context, such as sequence-to-sequence translation, named entity recognition, and sentiment analysis.
- **Robustness to noise and missing information**: Since BiLSTM models utilize information from both directions, they may be more robust to noise or missing information in the input sequence, as they can rely on context from both past and future elements to make predictions.
- **Enhanced representation**: The output representation of BiLSTM models is richer than that of standard LSTM models, as it incorporates hidden states from both the forward and backward LSTM layers. This can lead to improved performance when used as input to downstream layers or models.

In summary, Bidirectional Long Short-Term Memory (BiLSTM) models extend the standard LSTM architecture by processing input sequences in both forward and backward directions, allowing them to capture context from both past and future elements. This bidirectional processing offers several advantages, including improved context capture, better performance on complex tasks, robustness to noise and missing information, and enhanced output representation.

3.14 Transformer Cells

The Transformer model, introduced by Vaswani et al. in their 2017 paper "Attention is All You Need" [58], has revolutionized the field of natural language processing and sequence-to-sequence tasks. It is an innovative architecture that relies primarily on self-attention mechanisms and replaces the recurrent structure found in traditional RNNs and LSTMs. This section provides a detailed explanation of the Transformer model, its components, and how it works.

The Transformer model comprises two main parts: the encoder and the decoder. Both parts consist of a stack of identical layers, with each layer containing multi-head self-attention and position-wise feedforward sublayers. The architecture also incorporates residual connections and layer normalization to stabilize and accelerate training.

3.14.1 Encoder

The input sequence is first embedded into continuous vectors, followed by the addition of positional encoding. The positional encoding injects information about the position of each element in the sequence, as the Transformer model lacks inherent sequential processing capabilities.

Each encoder layer consists of two sublayers:

1. **Multi-head self-attention**: This sublayer computes attention scores between all pairs of elements in the input sequence. The attention mechanism allows the model to weigh the importance of each input element relative to others when encoding information. The self-attention mechanism is applied independently in parallel across multiple "heads," enabling the model to capture different aspects of the input sequence. The outputs of all attention heads are concatenated and passed through a linear transformation.

2. **Position-wise feedforward networks**: These are fully connected feedforward networks applied independently to each position in the sequence. They consist of two linear transformations with a ReLU activation function in between.

Residual connections and layer normalization are applied around both sublayers, which helps with gradient flow and model training.

3.14.2 Decoder

Similar to the encoder, the decoder consists of a stack of identical layers, each containing three sublayers:

1. **Masked multi-head self-attention**: This sublayer is similar to the multi-head self-attention in the encoder but includes a masking mechanism to prevent the decoder from attending to future elements in the output sequence. This ensures that the model maintains an autoregressive nature during training and inference.

2. **Encoder-decoder attention**: This sublayer computes attention scores between the output of the last encoder layer and the output of the previous decoder layer. This mechanism allows the decoder to focus on relevant parts of the input sequence when generating the output.

3. **Position-wise feedforward networks**: These are the same as in the encoder layers.

Residual connections and layer normalization are also applied around each sublayer in the decoder.

3.14.3 Training and Inference

The Transformer model is trained using a standard cross-entropy loss function, with the decoder's output at each time step being compared to the ground-truth target sequence.

During inference, the decoder generates output tokens one at a time in an autoregressive manner, using the previously generated tokens as input. Beam search or greedy search can be employed to find the most likely output sequence.

3.14.4 Key Advantages of Transformer Models

The Transformer model offers several advantages over traditional recurrent architectures, such as RNNs and LSTMs, which have been highlighted in the original paper by Vaswani et al. [58]:

- **Parallelization**: Unlike RNNs and LSTMs, which rely on sequential processing, the Transformer model can process input sequences in parallel, making it highly efficient on modern hardware.
- **Long-range dependencies**: The self-attention mechanism allows the model to capture long-range dependencies directly, without the need for recurrent connections.
- **Scalability**: The Transformer architecture scales well to large input sequences and has proven highly effective on various natural language processing tasks, including machine translation, language modeling, and summarization.

In summary, the Transformer model is a groundbreaking architecture that relies on self-attention mechanisms and parallel processing to address sequence-to-sequence tasks [58]. Its components, including multi-head self-attention, position-wise feedforward networks, residual connections, and layer normalization, contribute to its effectiveness and scalability in various natural language processing tasks [58]. Since its introduction, the Transformer architecture has laid the foundation for several state-of-the-art models, such as BERT, GPT, and T5, which have further pushed the boundaries in areas like language understanding, generation, and transfer learning [59, 60, 61].

Furthermore, the Transformer architecture has demonstrated its versatility by being adapted to non-textual domains such as computer vision, speech recognition, and reinforcement learning. The ability to capture long-range dependencies, parallelize computation, and scale effectively have made the Transformer model a cornerstone in the development of advanced deep learning techniques and applications.

3.14.5 Activation Calculations in Transformer Cells

The activation calculations in Transformer cells mainly occur in the multi-head self-attention sublayer and position-wise feedforward networks sublayer.

1. Multi-head Self-Attention Activation

The self-attention mechanism computes attention scores between all pairs of elements in the input sequence. For a given input sequence $X \in \mathbb{R}^{n \times d}$, where n is the sequence length and d is the dimension of the input embeddings, the self-attention mechanism can be defined as follows:

- Compute the query $Q = XW_Q$, key $K = XW_K$, and value $V = XW_V$ matrices, where $W_Q, W_K, W_V \in \mathbb{R}^{d \times d_k}$ are learnable weight matrices, and d_k is the dimension of the keys, queries, and values.
- Calculate the attention scores by taking the dot product of the query and key matrices and scaling by the square root of d_k: $S = \frac{QK^T}{\sqrt{d_k}}$.
- Apply a softmax function to the attention scores to obtain the attention probabilities: $A = \text{softmax}(S)$.
- Compute the output by multiplying the attention probabilities with the value matrix: $O = AV$.

In multi-head self-attention, this process is applied independently in parallel across multiple "heads," with the outputs of all attention heads concatenated and passed through a linear transformation.

2. Position-wise Feedforward Networks Activation

The position-wise feedforward networks are applied independently to each position in the sequence. Given an input $Z \in \mathbb{R}^{n \times d}$, where n is the sequence length and d is the dimension of the input embeddings, the position-wise feedforward networks can be defined as follows:

- Apply a linear transformation with weights $W_1 \in \mathbb{R}^{d \times d_f}$ and biases $b_1 \in \mathbb{R}^{d_f}$: $F_1 = ZW_1 + b_1$, where d_f is the dimension of the intermediate feedforward layer.
- Apply a ReLU activation function: $R = \text{ReLU}(F_1)$.
- Apply another linear transformation with weights $W_2 \in \mathbb{R}^{d_f \times d}$ and biases $b_2 \in \mathbb{R}^d$: $F_2 = RW_2 + b_2$.

The output of the position-wise feedforward networks, F_2, is then combined with the input Z through residual connections and layer normalization:

- Apply residual connections by adding the input Z to the output of the feedforward networks F_2: $R_c = Z + F_2$.
- Apply layer normalization to the residual output R_c: $L_n = \text{LayerNorm}(R_c)$.

This output, L_n, is then passed on to the next layer in the Transformer architecture.

It is important to note that the activations in the Transformer model are calculated in a parallel and efficient manner. The multi-head self-attention mechanism allows the model to focus on different aspects of the input sequence simultaneously, while the position-wise feedforward networks act independently on each position in the sequence. These features, combined with residual connections and layer normalization, contribute to the effectiveness and scalability of the Transformer model, enabling it to handle complex sequence-to-sequence tasks and scale well to large input sequences.

3.15 Datasets

This chapter provides an review of the datasets included and excluded in this study. First, the selected datasets will be described, highlighting their similarities and differences. Next, the omitted datasets will be presented in a table format, summarizing their characteristics.

3.15.1 Selected Datasets

The selected datasets for this study include electricity consumption data from different regions and countries, such as Spain, Paraguay, the United States, and Morocco. The datasets vary in terms of sampling frequency, data period, and the availability of additional data, such as weather and electricity pricing information. In the following sections, each dataset will be discussed in more detail.

Morocco

The Morocco, Tetouan dataset [62] features 10 months of electricity consumption data for three zones in the city of Tetouan, Morocco, with a sampling frequency of every 10 minutes. The data covers the entire year of 2017. The dataset also includes weather data such as temperature, humidity, and wind speed. Since the zones are in close proximity to one another, the weather data is assumed to be representative of all three zones.

Paraguay

The Paraguay dataset [63] features 48 months of electricity consumption data for the Alto Paraná region in Paraguay, sampled at an hourly frequency. The data covers the period from January 2017 to January 2021. Additionally, the dataset includes weather data such as temperature, humidity, wind speed, and pressure, measured every three hours.

The weather data was collected by a single weather station, while the energy consumption data was recorded from 55 feeders across 15 different substations. By aggregating the data from these feeders, the total energy consumption for the region can be determined.

Spain Cities

The Spain Cities dataset [64] contains 48 months of electricity consumption data from January 2015 to January 2019 for Spain, sampled at a frequency of 1 hour. Accompanying the electricity production data for Spain, is weather data for the five largest cities in Spain: Madrid, Barcelona, Valencia, Seville, and Bilbao. The weather data comprises temperature for each timestamp.

The electricity consumption and generation data were sourced from a public portal called ENTSOE, while the pricing data was gathered from the Spanish Transmission System Operator, Red Eléctrica de España. The weather data was purchased from the Open Weather API for the aforementioned cities. This comprehensive dataset has been made publicly available on Kaggle.

Spain Houses

The Spain Houses dataset [65] comprises 12 months of electricity consumption data for 499 smart meters in a specific region in Spain, with a sampling frequency of 1 hour. The data covers the entire year of 2019. Each of the 499 customers is assigned one of 68 different customer profiles, such as private households, shops, and bakeries. Furthermore, the dataset contains outdoor temperature data for each location at every timestamp.

USA

The USA dataset [66] includes 55 months of electricity consumption data for 20 zones, sampled at an hourly frequency. The data spans from January 2004 to June 30, 2008. The dataset also contains temperature data from 11 different weather stations for each hour. However, there is insufficient information to determine the correspondence between electricity zones and weather stations. Therefore, the electricity and weather data can only be meaningfully combined when aggregated and averaged for each timestamp.

3.15.2 Similarities in Datasets

All selected datasets share certain common characteristics, which make them suitable for comparative analysis in this study. These commonalities include:

- **Historical Electricity Consumption:** Each dataset contains historical electricity consumption or load data, providing valuable information on energy usage patterns over time.
- **Temperature Data:** All datasets include temperature data, allowing for the investigation of potential correlations between weather conditions and electricity consumption.
- **Sampling Frequency:** Except for the Paraguay weather data, all datasets have a sampling frequency of one hour or faster for both electricity and weather features, ensuring a consistent level of granularity across datasets.
- **Aggregated Consumption Data:** All datasets, except for the Spain smart meter data, provide consumption data aggregated across multiple houses and businesses. The Spain smart meter data can be easily aggregated during the data processing step, ensuring consistency across datasets.

3.15.3 Differences in Datasets

While the selected datasets share common features, they also exhibit several differences in terms of data collection, reporting methodologies, and geographical coverage. Some of the key differences include:

- **Weather Data Granularity:** There is a significant variation in the granularity of the accompanying weather data across datasets. For instance, the Spain smart meter dataset provides temperature measurements in close proximity to the electricity consumption locations. While this can potentially lead to stronger correlations between weather and consumption, the measurements may be of lower quality due to local factors such as open windows, heat reflection from buildings, or sensor placement. Conversely, the Spain cities dataset relies on weather data from the five largest cities to cover an area exceeding half a million square kilometers, which may affect the accuracy of weather-consumption correlations.
- **Geographical Coverage:** The datasets cover various parts of the world, including North America, South America, Europe, and Northern Africa. These regions exhibit different seasonal patterns and weather conditions, which can influence electricity consumption behaviors. Moreover, social and cultural factors may also contribute to variations in electricity usage across these locations.
- **Data Collection and Reporting:** The datasets differ in their data collection and reporting methodologies, which can impact the comparability of results. For example, the Paraguay dataset relies on weather data collected from a single station, while the USA

dataset includes data from multiple weather stations but lacks sufficient information to determine the relationship between electricity zones and weather stations.

- **Consumer Profiles:** The datasets also vary in terms of the consumer profiles represented, which can affect electricity consumption patterns. For instance, the Spain Houses dataset assigns customers to one of 68 different profiles, such as private households, shops, and bakeries. This diversity in consumer profiles can result in a wide range of consumption behaviors and responses to weather conditions.

In summary, the selected datasets for this study share several similarities in terms of historical electricity consumption data, temperature data, sampling frequency, and aggregated consumption data. However, they also exhibit differences in weather data granularity, geographical coverage, data collection and reporting methodologies, and consumer profiles. These differences need to be carefully considered when analyzing the datasets and drawing conclusions from the results.

3.15.4 Omitted Datasets

During the process of reviewing literature and searching for suitable datasets, a total of 15 potential candidates were identified. All of these datasets were publicly available on the internet and contained measurements spanning at least six months. The datasets are listed in Table 1, along with information regarding the availability of weather data, sampling frequency, time length of the data, perceived difficulty of extracting data, and additional notes on the content. The first five entries in the table are the datasets that were ultimately chosen for this study.

The first omitted dataset contains electricity consumption data for a single building in France. This dataset was not considered a suitable candidate for the transferability task, as it represents a single building rather than an aggregation of consumption data from multiple households and industries, which is the case for the chosen datasets.

The remaining omitted datasets were excluded primarily due to the absence of accompanying weather data. The decision was made not to use a weather API to gather this data for several reasons. Firstly, the different datasets exhibit considerable variation in geographical size and sampling frequency, which would complicate the process of obtaining accurate and representative weather data. Secondly, the addition of weather data from an API would substantially increase the number of data points to be trained on and evaluated against in the transferability study, introducing additional complexity and computational requirements.

Data source	Notes	Weather data	Frequency	Time length	Difficulty in extracting
Spain cities [64]	Weather data from 5 Largest cities in spain	Yes	1 Hour	4 Years	Easy
Morocco, Tetouan [62]	Power from 3 zones in city	Yes	10 Minutes	10 Months	Easy
Spain houses [65]	500 Household smartmeters + sensors in spain	Yes	1 Hour	1 Year	Medium
Paraguay [63]	1 Region in Paraguay	Yes	1 Hour	4 Years	Easy
USA + Weather [66]	US utility data in 20 zones	Yes	1 Hour	4 Years	Easy
France [67]	1 Building (Challenger building)	Yes	10 Minutes	2.4 Years	Easy
Belgium [68]	Whole belgium consumption	No	15 minutes	7 Years	Easy
UK-London [69]	5500 different energy measuring units accorss city.	No	30 Minutes	2.4 Years	Easy
USA, no weather [70]	Partially covers 14 states, but changes some regions over time	No	1 Hour	14 Years	Easy
EU [71]	Data from 35 EU countries	No	1 Hour	5 - 24 Years	Easy
France 1 house [72]	Data from 1 household	No	1 Minute	4 Years	Easy
NYSIO [73]	Regions in New York	No	1 Hour	4 Years	Easy
Australia [74]	Messy data of Australian historical demand	No	5 Minutes - 1 Hour	10+ Years	Medium, scraping and combining different files
USA TMY3 [75]	~1000 cities + many building types	No	1 Hour	1 Year	Hard, many different files and structures
CNU [76]	Smartmeters University buildings	No	1. Hour	1.3 Years	Very Hard, bad structure + Korean language

Table 1: Datasets considered for analysis

3.16 Data pre-processing

The data preprocessing step focuses on transforming the various datasets into a common format, enabling them to be processed by the same models, as well as cleaning any unrealistic data. The primary goal is to ensure that all datasets have two columns: Consumption and Temperature. The transformations applied to each dataset are described below.

3.16.1 Morocco Dataset

For the Morocco dataset, the electricity consumption from the three different zones was aggregated into a single column. This aggregation allowed for a unified representation of electricity consumption across the zones, providing a more comprehensive view of the data.

3.16.2 Paraguay Dataset

In the case of the Paraguay dataset, the electricity consumption data from all 55 station sub-feeders were combined to form a single consumption column. Additionally, the last 360 hours of data were removed due to missing consumption values, ensuring the dataset's consistency and reliability.

3.16.3 Spain Cities Dataset

For the Spain Cities dataset, a master dataset was created representing the combined electricity consumption for the entire country. Furthermore, individual datasets were generated for each of the five largest cities in Spain (Valencia, Bilbao, Barcelona, Madrid, and Seville). In these city-specific datasets, each city retained its respective weather data, but the consumption data was adjusted using the formula: Spain consumption * (population percentage of the city compared to the sum of the population of the five cities). This approach ensured that the city-level datasets accurately reflected the electricity consumption magnitudes for each city.

3.16.4 Spain Houses Dataset

In the Spain Houses dataset, the electricity consumption of all 499 smart meters was aggregated into a single column, and the Temperature column was set to the average temperature recorded by all the smart meters. Approximately 300 outlier consumption points were identified and grouped together. These outliers were treated as missing values to ensure the dataset's integrity and prevent potential distortions in the analysis.

3.16.5 USA Dataset

The USA dataset's format was transformed from a structure where each row represented a day with 24 columns (one for each hour) to a format consistent with the other datasets, including TimeStamp, Consumption, and Temperature columns. Moreover, 18 missing values at the tail end of the dataset were removed, ensuring data continuity and consistency.

3.16.6 Handling Missing Values

After transforming all the datasets into a consistent format and ensuring that no missing values were present at the beginning or end of each dataset, cubic spline interpolation was employed to fill in any remaining missing values. Cubic spline interpolation provides a smooth curve that fits the data points and helps maintain the time series' continuity while addressing missing values. This approach ensures that the datasets are complete and suitable for subsequent analysis using machine learning models.

3.17 Training and Testing setup

3.17.1 48 hour lookback horizon

The autoregressive approach is a widely-accepted method in time series analysis, particularly in electricity demand forecasting. This technique leverages the historical values of a time series to predict its future values. In the context of this study, a lookback horizon of 48 hours is employed for the purposes of hyperparameter tuning and testing different features, which translates to using the previous 48 hourly electricity demand data points as input features for predicting future demand. The rationale behind the choice of a 48-hour lookback horizon stems from the following factors:

Temporal dependencies: Electricity demand exhibits strong temporal dependencies, with patterns recurring daily, weekly, and seasonally. These patterns arise from various factors, such as human activities, differences in energy consumption between weekdays and weekends, weather conditions, daylight hours, and holidays.

A 48-hour lookback horizon allows the model to capture these patterns effectively, particularly the daily fluctuations in demand, by incorporating data from the previous two days. This approach enables the model to learn from the immediate past as well as from the same time of day from the day before. Consequently, the model can better account for daily and weekly variations in electricity demand, which is critical for accurate forecasting.

In addition to daily, weekly, and seasonal patterns, electricity demand also exhibits other temporal dependencies that influence forecasting accuracy. These include intraday patterns, special events and holidays, and weather-related dependencies.

Intraday patterns are distinct variations in demand within a single day, reflecting changes in human activities, business operations, and weather conditions. A 48-hour lookback horizon captures these intraday patterns by incorporating data from multiple periods throughout the day, allowing the model to better anticipate fluctuations in demand that may occur within a 24-hour cycle.

Special events and holidays can significantly affect electricity demand, leading to atypical consumption patterns that differ from the typical daily, weekly, or seasonal trends. The 48-hour lookback horizon can hopefully provide the models with enough historical context to detect and adjust for such deviations in demand when they arise, enhancing its ability to predict electricity consumption accurately during these periods.

Computational efficiencies: Using a 48-hour lookback horizon offers several advantages in terms of computational efficiency compared to longer lookback horizons, such as one and two weeks. These advantages include Reduced input dimensionality and Faster convergence:

As the lookback horizon increases, the number of input features also increases proportionally. For instance, a one-week lookback horizon would require 168 input features (24 hours x 7 days), while a two-week lookback horizon would require 336 input features (24 hours x 14 days). In contrast, a 48-hour lookback horizon only requires 48 input features, considerably reducing the input dimensionality.

This reduction in input dimensionality translates to fewer parameters that the model needs to learn, which in turn leads to faster training times and less memory consumption. Furthermore, a lower-dimensional input space can help alleviate the curse of dimensionality, a phenomenon where models struggle to learn effectively as the feature space becomes increasingly sparse and complex.

With fewer input features, the optimization process during model training becomes more manageable, allowing the model to converge to a suitable solution faster. A more rapid convergence reduces the overall time required to train and fine-tune the model, making it more feasible to experiment with different hyperparameters and model architectures in the search for an optimal configuration. In practical applications of electricity demand forecasting, models often need to be updated and retrained frequently to incorporate the most recent data and maintain their predictive accuracy. A 48-hour lookback horizon allows for faster model updates and retraining, making it more suitable for real-time forecasting and adaptive systems.

Generalizability: An important aspect of model performance in electricity demand forecasting is generalizability, which refers to the model's ability to perform well on new, unseen data from different regions, time periods, or energy consumption profiles. The choice of a 48-hour lookback horizon positively impacts generalizability in several ways.

By using a 48-hour lookback horizon, the model prioritizes the most recent and relevant data for predicting future electricity demand. This allows the model to adapt quickly to any changes in consumption patterns, such as shifts in human activities, technological advancements, or policy interventions. By focusing on the most pertinent information, the model can better generalize to new datasets and different time periods, as it learns to capture the underlying dynamics of electricity demand rather than memorizing historical patterns specific to the training dataset.

As mentioned earlier, shorter lookback horizons can help mitigate the risk of overfitting, which occurs when the model learns to perform exceptionally well on the training data but struggles to generalize to unseen data. By limiting the input features to the most recent 48 hours, the model is less likely to overfit to specific training data and can better generalize to different contexts.

A 48-hour lookback horizon enables the model to quickly adapt to local variations in electricity demand patterns. This is particularly important when forecasting electricity consumption in different regions, as local factors such as climate, infrastructure, and socio-economic characteristics can significantly influence energy consumption patterns. By focusing on the most recent data, the model can learn the unique characteristics of each region, allowing it to generalize effectively across diverse geographical locations.

Shorter lookback horizons, such as 48 hours, make the model more robust to data anomalies or outliers that may be present in the historical data. These anomalies can result from measurement errors, data processing issues, or atypical events that are not representative of the underlying demand patterns. By not relying heavily on data from the distant past, the model can better filter out the impact of these anomalies and maintain its generalizability across different datasets and scenarios. And having a longer lookback horizons means that anomalies will be a part of a larger percent of the training examples.

A model trained with a 48-hour lookback horizon may also have greater potential for transfer learning. The 48-hour lookback horizon enables the model to capture essential temporal dependencies without overfitting to specific historical patterns, making it a suitable starting point for transfer learning across different contexts.

3.17.2 Longer lookback horizons

While a 48-hour lookback horizon offers numerous advantages in terms of computational efficiency and generalizability, there are situations where employing longer lookback horizons, such as 168 hours (one week) or 336 hours (two weeks), may yield benefits for specific machine learning models or forecasting tasks. The potential benefits of these extended lookback horizons encompass several aspects:

Uncovering extended temporal relationships: Longer lookback horizons can reveal crucial extended temporal dependencies in the data, which might be vital for certain forecasting tasks or specific energy consumption patterns. For instance, industrial electricity demand may exhibit multi-week patterns influenced by production cycles, maintenance schedules, or supply chain dynamics. A one or two-week lookback horizon would enable the model to recognize and learn from these broader patterns, enhancing forecasting accuracy in such situations.

Recognizing weekly and seasonal trends: A more extended lookback horizon can supply the model with a richer context for identifying and learning weekly and seasonal patterns in electricity demand. With access to data from multiple weekdays and weekends within a one or two-week lookback, the model can better distinguish and adapt to the unique consumption patterns associated with each. Furthermore, accurately forecasting seasonal patterns might necessitate a more extensive context, particularly in areas with substantial weather variations or fluctuating daylight hours.

Incorporating external influences and events: Extended lookback horizons may prove beneficial when external factors such as holidays, special events, or extreme weather conditions have a lasting impact on electricity demand. By examining data from the previous one or two weeks, the model can learn how these external factors affect electricity consumption over an extended period, allowing it to better anticipate and adjust its forecasts for similar events in the future.

Facilitating enhanced model flexibility: Longer lookback horizons offer more flexibility in model architecture and design. For example, recurrent neural networks (RNNs) and their variations, such as long short-term memory (LSTM) and gated recurrent unit (GRU) networks, can effectively learn long-term dependencies with extended sequences of data. These architectures might perform better with a one or two-week lookback horizon, as they can model more complex temporal relationships in the data.

Employing ensemble approaches and model stacking: In some cases, integrating the forecasts from models with varying lookback horizons can lead to improved overall performance. For instance, a model with a 48-hour lookback horizon might excel at capturing short-term fluctuations, while a model with a one or two-week lookback horizon could better account for longer-term trends and cyclical patterns. By assembling an ensemble or stacking these models, the strengths of each can be combined to produce more accurate and robust forecasts.

3.17.3 Hyperparameter tuning

This section presents the methodology for setting up the training and testing environment, focusing on the hyperparameter tuning process. The aim was to give all models an equal opportunity to perform well during optimization, tailoring the tuning process for each specific model. The models and their respective tunable parameters are as follows:

- Ridge Regression: alpha parameter
- Gradient Boosting Model: number of estimators and number of leaves
- LSTM and BiLSTM: internal model lookback length and number of layers
- Transformer: number of blocks and number of heads

To optimize the hyperparameters, a grid search optimization method was employed. In this approach, each model's tunable parameters were assigned a selection of values to be explored during the optimization process. The Ridge Regression and Gradient Boosting Models had a wider range of possible parameter combinations, as they are relatively quick to train. In contrast, the LSTM, BiLSTM, and Transformer models had their parameter selections adjusted to ensure that the tuning process could be completed within 12 hours on an NVIDIA P100 graphics card.

The grid search optimization method consists of exhaustively trying every possible combination of the selected parameter values. This systematic approach allows for the identification of the best-performing hyperparameter configuration for each model. It is important to note that the grid search optimization method can be computationally expensive, especially for complex models such as the LSTM, BiLSTM, and Transformer. Therefore, careful consideration was given to the selection of parameter values for these models to ensure the optimization process remained feasible within the given computational constraints.

Given that the problem deals with time series data, cross-validation could not be used. Instead, a train-test split approach was employed, where the test set comprised the last 90 days of the dataset. This method maintains the temporal order of the data and ensures that the models are evaluated on their ability to predict future observations.

3.17.4 Training and Testing Parameters

This section discusses the parameters used during the training and testing loop for the LSTM, BiLSTM, and Transformer models. These parameters have been carefully selected to optimize the performance of the models and ensure efficient training.

Batch Size A batch size of 32 was used for the LSTM, BiLSTM, and Transformer models. The choice of this batch size strikes a balance between computational efficiency and the ability to generalize from the training data. A larger batch size may lead to faster training, but it could also result in less accurate generalization, while a smaller batch size may improve generalization at the cost of increased training time.

Training Epochs and Learning Rate The three deep learning models were trained for a maximum of 150 epochs, with learning rate schedulers on plateau to adjust the learning rates dynamically based on the training progress. This approach aimed to maximize the performance of the models by allowing them to converge to optimal solutions within a reasonable number of training iterations.

For the LSTM and BiLSTM models, the initial learning rate was set to 0.001. The Transformer model had a slightly higher initial learning rate of 0.002. The difference in the learning rates can be attributed to the distinct architectures and training dynamics of the models, with the Transformer potentially benefiting from a higher initial learning rate due to its parallel and self-attention-based architecture.

Data Preprocessing Before training and testing, the data was preprocessed using Scikit-learn's StandardScaler. This step standardized the features by removing the mean and scaling them to unit variance. Standardization is crucial in machine learning models, as it ensures that all features have the same scale, preventing any feature from dominating others due to differences in their magnitude. This preprocessing step contributed to a more stable and efficient training process and improved the overall performance of the models.

3.17.5 Metrics

Mean Absolute Error (MAE):

Mean Absolute Error (MAE) is a commonly used metric to evaluate the performance of regression models. It measures the average magnitude of the errors in a set of predictions, without considering their direction. MAE is calculated as the average of the absolute differences between the predicted values and the actual values.

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

Where n is the number of samples, y_i is the actual value, and $\hat{y}_i$ is the predicted value for sample i.

Root Mean Squared Error (RMSE):

Root Mean Squared Error (RMSE) is another popular metric used to evaluate regression models. RMSE measures the average squared difference between the predicted values and the actual values. By squaring the differences, RMSE gives a higher weight to larger errors. The square root of the mean squared error is then taken to bring the values back to the original scale.

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

Where n is the number of samples, y_i is the actual value, and $\hat{y}_i$ is the predicted value for sample i.

Mean Absolute Percentage Error (MAPE):

Mean Absolute Percentage Error (MAPE) is a metric that expresses the prediction errors as a percentage of the actual values. MAPE is particularly useful when dealing with data that has different scales, as it allows for a more interpretable comparison of errors across different datasets.

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

Where n is the number of samples, y_i is the actual value, and $\hat{y}_i$ is the predicted value for sample i.

When working with electricity forecasting datasets that have different scales for electricity usage, using MAPE is the most sensible choice. This is because MAPE expresses errors as a percentage of the actual values, allowing for a more interpretable comparison of the model's performance across different datasets, regardless of their scale. Moreover, it enables a better understanding of the relative magnitude of the errors, which can be crucial in evaluating the model's effectiveness in forecasting electricity consumption.

4 Results

In the subsequent results section, the findings from the study are presented and analyzed. The primary focus of this research is to compare the performance, generalizability, and explainability of transformer-based models with both simple machine learning methods, such as Ridge Regression and Gradient Boosting Models, and more advanced deep learning methods, including Long Short-Term Memory (LSTM) and Bidirectional LSTM (BiLSTM) networks.

Initially, the performance of these models is compared under various parameter settings and feature combinations. Following the performance analysis, the generalizability and explainability of the models are assessed to better understand their practical implications and applicability in real-world settings. By providing a comprehensive evaluation of these methods, the results aim to offer valuable insights into the strengths and weaknesses of each approach, ultimately contributing to a better understanding of the most suitable techniques for addressing the research question at hand.

4.1 Descriptive statistics

In this section, the descriptive statistics for the different datasets utilized in the study are presented with Table 2. The datasets include information on electricity consumption and temperature for several locations, including Morocco, Paraguay, Spain, and the United States. The key statistics include the count, mean, standard deviation, minimum, 25%, 50%, and 75% quantiles, and the maximum values for both consumption and temperature.

Table 2: Descriptive statistics for the datasets

Morocco

Statistic	Consumption	Temperature
count	8736	8736
mean	427337.3	18.8
std	102106.0	5.8
min	224445.6	3.6
25%	339512.28	14.4
50%	419430.0	18.8
75%	502689.8	22.9
max	798779.0	39.7

Paraguay

Statistic	Consumption	Temperature
count	34704	34704
mean	5587.5	22.6
std	1183.2	6.0
min	2064.8	-0.2
25%	4735.4	19.0
50%	5505.9	22.8
75%	6414.7	26.4
max	9550.1	40.8

Spain Cities Average

Statistic	Consumption	Temperature
count	35064	35064
mean	28698.4	16.5
std	4576.0	7.3
min	18041.0	-1.2
25%	24807.8	10.9
50%	28902.0	15.9
75%	32194.0	21.9
max	41015.0	36.1

Spain Houses

Statistic	Consumption	Temperature
count	8760	8760
mean	718.1	17.8
std	1047.3	6.1
min	413.4	3.7
25%	631.9	12.9
50%	790.5	17.6
75%	988.1	22.4
max	4376.1	33.6

USA

Statistic	Consumption	Temperature
count	39414	39414
mean	1618269	14.1
std	476383.9	9.6
min	836931	-13.6
25%	1373222	6.6
50%	1579918	14.7
75%	1885758	21.9
max	3280423	37.9

4.2 Ridge

In the Ridge Regression subsection of the Results, several aspects of the model's performance are examined, including the impact of different lookback horizons, hyperparameter tuning, incorporation of lagged temperature and time encodings, and the addition of other features.

The first analysis explored the effect of varying lookback horizons on the performance of the Ridge Regression model. The model was tested with 48-hour, 168-hour, and 336-hour lookback horizons. The results in Figure 1a indicate a clear trend where longer lookback horizons lead to improved performance.

The second aspect of the analysis involved tuning the alpha parameter, which is a crucial hyperparameter in Ridge Regression. From Figure 1b, adjusting the alpha parameter did not lead to any noticeable changes in the model's performance. This result indicates that the Ridge Regression model is relatively robust to changes in the regularization strength.

The third part of the analysis focused on incorporating lagged temperature and time encodings as input features. For the Paraguay dataset, the addition of these features resulted in a slight improvement in the model's performance as seen in Figure 1c. However, for the Spain Houses dataset, the performance slightly deteriorated. Overall, the inclusion of lagged temperature and time encodings did not lead to a substantial improvement in the Ridge Regression model's performance.

Lastly, the impact of adding other features, such as target values without cyclicality and data augmentation, was investigated and results are shown in Figure 1d. Including target values without cyclicality yielded marginally better results compared to the baseline model. However, data augmentation led to a decrease in performance compared to the baseline.

In summary, the Ridge Regression model showed improved performance with longer lookback horizons but demonstrated limited sensitivity to hyperparameter tuning and the inclusion of additional features. These findings provide valuable insights into the model's performance characteristics.

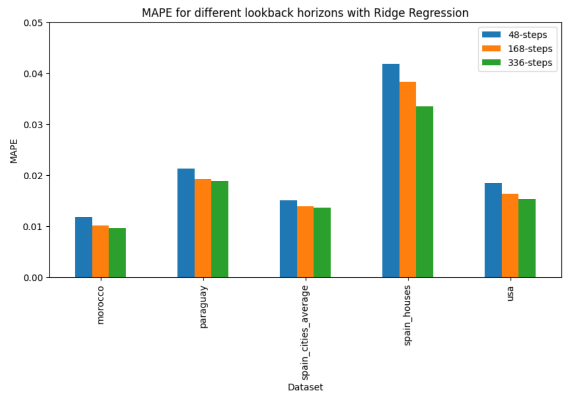
4.3 GBM

The first analysis in Figure 2a examined the effect of varying lookback horizons on the performance of the GBM . There was no clear pattern indicating whether longer lookback horizons were consistently better. For the Spain Houses dataset, longer lookback horizons showed improved performance, while for the Morocco and Spain Cities datasets, longer lookback horizons led to worse performance.

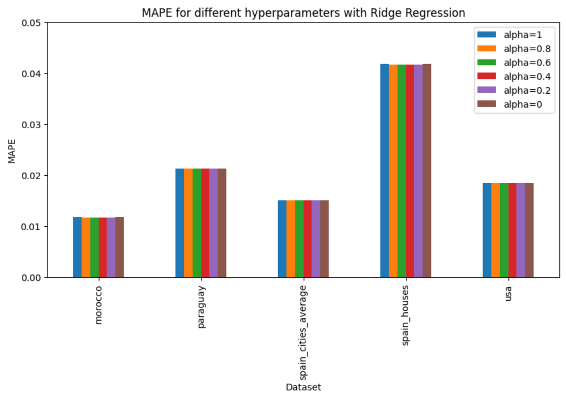
The second aspect of the analysis involved tuning the hyperparameters, specifically the number of estimators and the number of leaves. The results of this iw presented in Figure 2b. In general, more estimators and more leaves resulted in better performance. However, there was a trade-off with computational complexity, as increasing these factors also increased the computational time. Among the combinations that performed similarly well, the combination of 300 estimators and 100 leaves was selected for further studies.

The third part of the analysis found in Figure 2c focused on incorporating lagged temperature and time encodings as input features. The inclusion of these features led to slight improvements in the performance of the GBM model for the Paraguay, Spain Houses, and USA datasets.

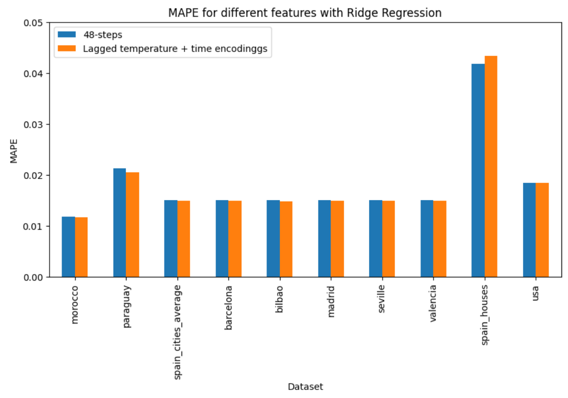
Lastly, the impact of adding other features, such as data augmentation and target values without cyclicality, was investigated and is shown in Figure 2d. Data augmentation had a



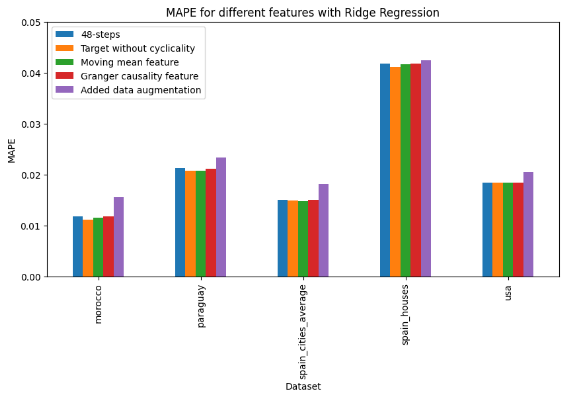
(a) Lookback horizon Ridge MAPE



(b) Hyperparameter tuning Ridge



(c) Weather feature for Ridge



(d) Other features with Ridge

Figure 1: Ridge model evaluation

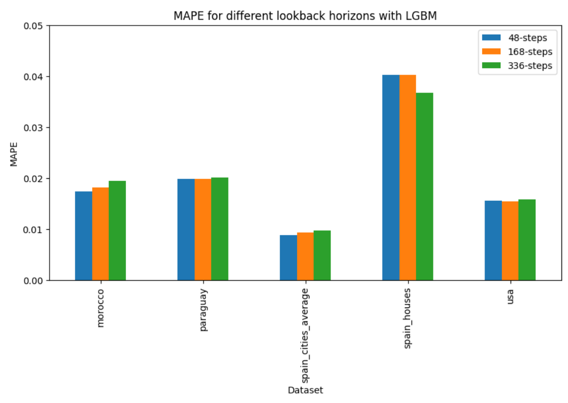
significant positive impact on the performance of the GBM model. The inclusion of target values without cyclicality resulted in a slight improvement for the Spain Houses dataset.

In summary, the GBM model's performance varied with different lookback horizons, and there was a trade-off between performance and computational complexity when tuning hyperparameters. The addition of lagged temperature and time encodings led to slight improvements for some datasets, while data augmentation showed a substantial improvement in the model's performance.

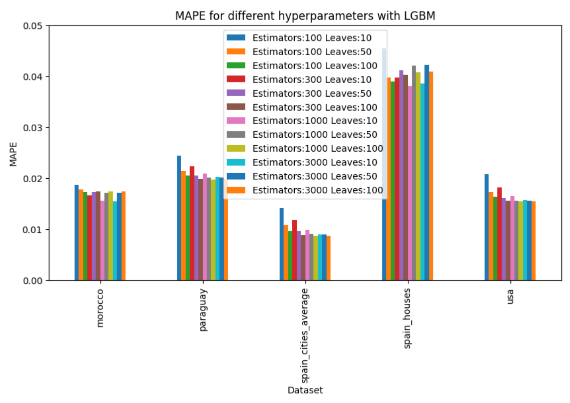
4.4 LSTM

In the "LSTM" section of the Results, the performance of the Long Short-Term Memory (LSTM) model is evaluated in terms of its ability to predict electricity consumption patterns. The primary metric used for evaluation is the Mean Absolute Percentage Error (MAPE), which allows for a fair comparison of the models' prediction capabilities.

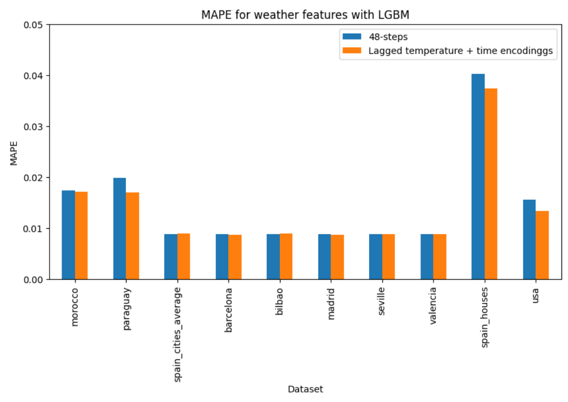
The LSTM model's performance appears to be negatively affected by an increase in the number of lagged consumption steps sent in as features as shown in Figure 3a. Specifically, the model's performance tends to worsen on the Morocco, Paraguay, and Spain Cities datasets,



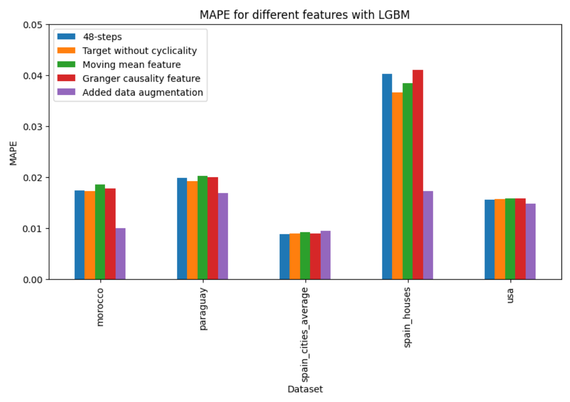
(a) Lookback horizon GBM MAPE



(b) Hyperparameter tuning GBM



(c) Weather feature for GBM



(d) Other features with GBM

Figure 2: GBM model evaluation

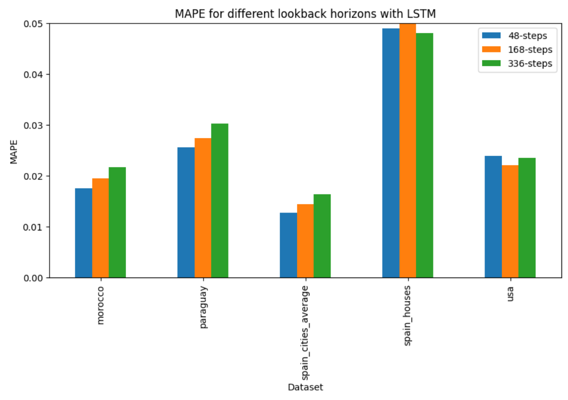
while demonstrating no consistent pattern on the other two datasets.

Regarding hyperparameter tuning, the internal number of lookback steps in the LSTM model and the number of layers were the primary focus. From Figure 3b it is observable that with only 24 hours of internal lookback and two layers of 32 neurons, the model exhibited exploding gradients, indicating potential instability. To mitigate this issue, the selected parameters for further testing included a lookback of 6 hours and two layers of LSTM blocks.

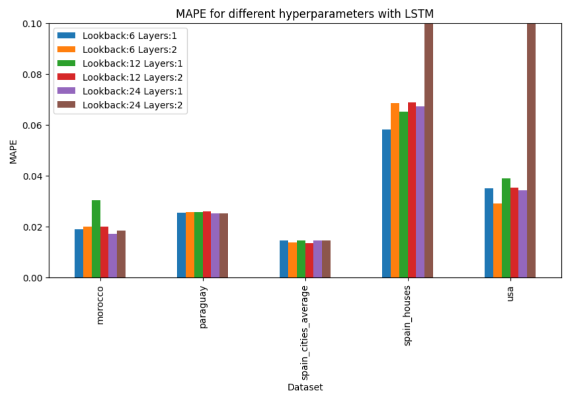
When temperature and time encoding were included as features as shown in Figure 3c, a consistent decrease in the model's performance was observed in 8 out of 10 cases. This finding suggests that incorporating these additional features may not be beneficial for the LSTM model in the context of electricity consumption forecasting.

As presented in Figure 3d, the introduction of data augmentation appeared to enhance the performance of the LSTM model for all datasets except Spain Cities. Furthermore, the inclusion of the "Target without cyclicality" feature resulted in minor improvements for the Spain Houses and USA datasets.

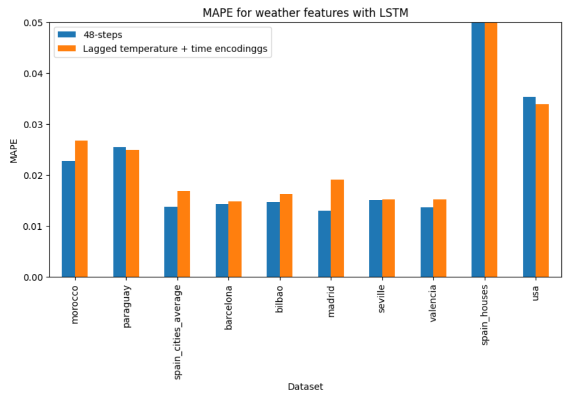
In conclusion, the LSTM model's performance in predicting electricity consumption patterns seems to be influenced by various factors, including the number of lagged consumption steps, hyperparameters, and additional features. While the model exhibits some sensitivity to these factors, the results also highlight potential areas for further investigation and optimization.



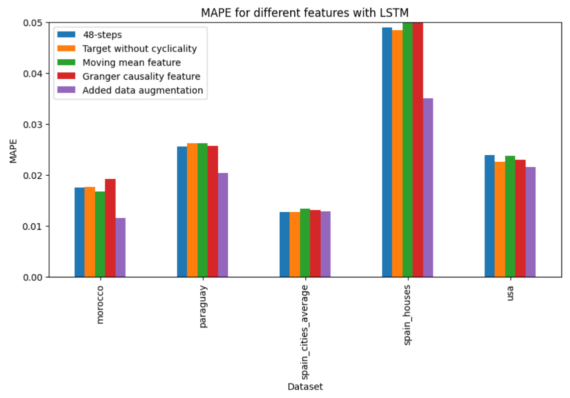
(a) Lookback horizon LSTM MAPE



(b) Hyperparameter tuning LSTM. The Lookback:24 Layers:2 setting gave a MAPE of 0.145 for spain_houses and 0.638 for usa dataset.



(c) Weather feature for LSTM. The baseline for spain_houses gave a MAPE of 0.062 and the model with lagged temperature and time encoding gave a MAPE of 0.147



(d) Other features with LSTM

Figure 3: LSTM model evaluation

4.5 BiLSTM

In the "BiLSTM" section of the Results, the performance of the Bidirectional Long Short-Term Memory (BiLSTM) model is assessed in terms of its ability to predict electricity consumption patterns. The primary metric used for evaluation is the Mean Absolute Percentage Error (MAPE), which provides a fair basis for comparing the prediction capabilities of the models.

According to Figure 4a The BiLSTM model's performance does not appear to improve with an increase in the number of lagged consumption steps sent in as features. In fact, the model's performance tends to worsen on all datasets, indicating that the model may not benefit from additional lagged steps.

Regarding hyperparameter tuning presented in Figure 4b, the internal number of lookback steps in the LSTM model and the number of layers were the primary focus. With just 24 hours of internal lookback and one or two layers of 32 neurons, the model exhibited exploding gradients, suggesting potential instability. To address this issue, the selected parameters for further testing included a lookback of 6 hours and two layers of BiLSTM blocks.

When temperature and time encoding were included as features, a consistent decrease in the model's performance was observed in 9 out of 10 cases presented in Figure 4c. The only exception was the USA dataset, where the model's performance did not exhibit a similar decline.

The introduction of data augmentation appeared to enhance the performance of the BiLSTM model across all datasets. However, none of the other features seemed to yield any significant improvements in the model's performance as can be seen in Figure 4d.

In conclusion, the BiLSTM model's performance in predicting electricity consumption patterns is influenced by various factors, including the number of lagged consumption steps, hyperparameters, and additional features. While the model demonstrates sensitivity to these factors, the results also highlight potential areas for further investigation and optimization to improve the model's performance in this domain.

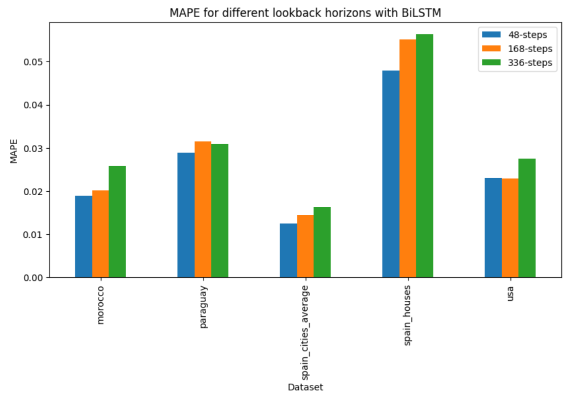
4.6 Transformer

The first analysis examined the effect of varying lookback horizons on the performance of the Transformer model. From Figure 5a it is observable that increasing the lookback horizon from 48 steps led to improvements for the Morocco, Paraguay, and Spain Houses datasets. However, there was no noticeable improvement for the USA dataset and a slight regression for the Spain Cities dataset.

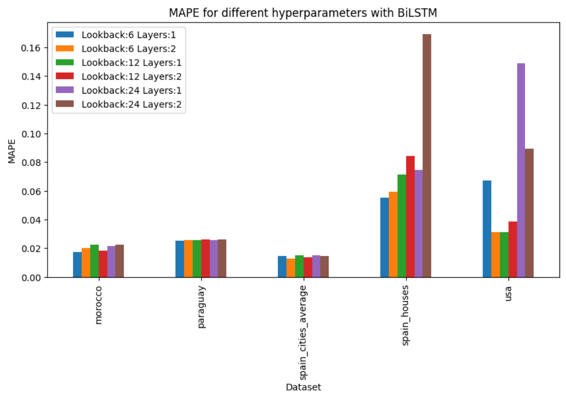
The second aspect of the analysis is presented in Figure 5b and involved tuning the hyperparameters, specifically the number of Transformer blocks and the number of heads for each block. Given the ease of overfitting on these datasets compared to image or text data, the grid search was set to relatively small parameters. The hyperparameter tuning results exhibited high variance, with some parameters performing well on certain datasets and poorly on others. On an overall basis, the combination of 2 blocks and 8 heads per block yielded the best average performance and was selected for further studies.

The third part of the analysis focused on incorporating lagged temperature and time encodings as input features. From the results presented in Figure 5c, most datasets experienced slight improvements with the inclusion of these features. However, there were slight regressions for the Morocco and Seville datasets, and no difference in performance for the Valencia dataset.

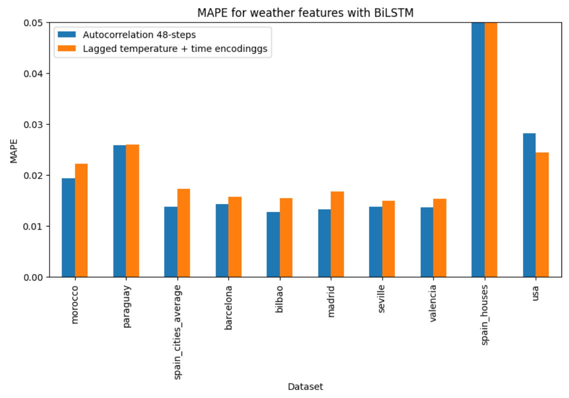
Lastly, the impact of adding other features, such as target values without cyclicality and the Granger causality feature, was investigated in Figure 5d. Including target values without



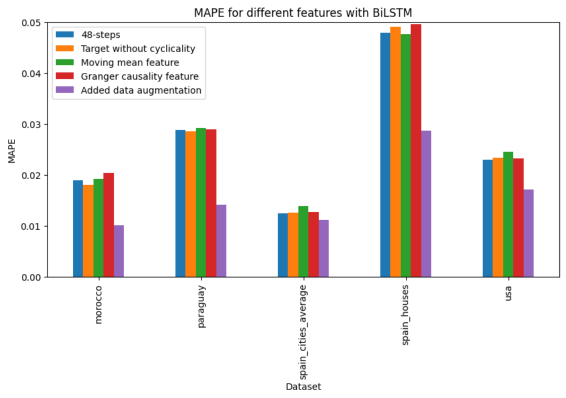
(a) Lookback horizon BiLSTM MAPE



(b) Hyperparameter tuning BiLSTM.



(c) Weather feature for BiLSTM. The baseline for spain_houses gave a MAPE of 0.062 and the model with lagged temperature and time encoding gave a MAPE of 0.407



(d) Other features with BiLSTM

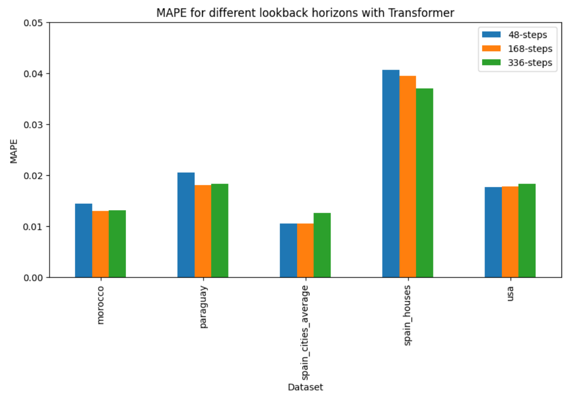
Figure 4: BiLSTM model evaluation

cyclicality improved performance in most cases. The Granger causality feature performed significantly better on the Morocco dataset but significantly worse on the Spain Houses dataset.

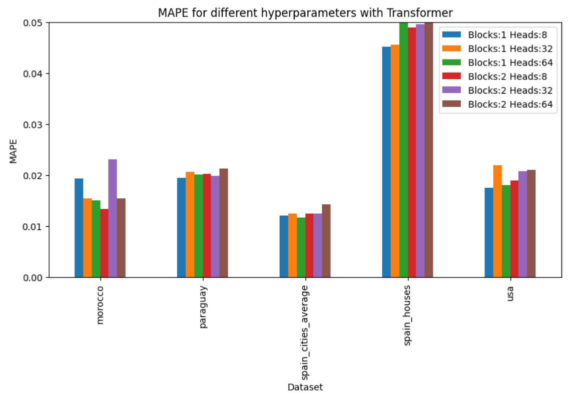
In summary, the Transformer model's performance varied with different lookback horizons and showed high variance in hyperparameter tuning results. The addition of lagged temperature and time encodings led to slight improvements for most datasets, while the inclusion of other features, such as target values without cyclicality and the Granger causality feature, produced mixed results.

4.7 Comparing models on lookback horizons

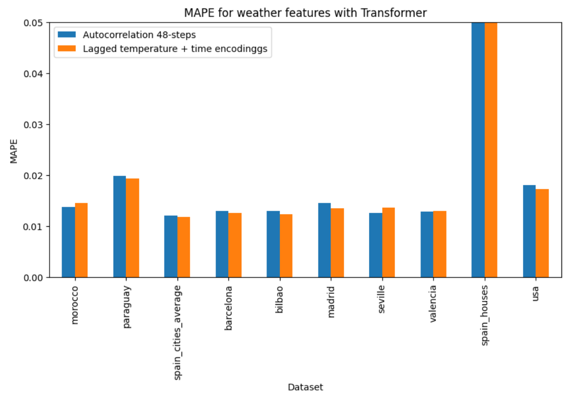
In the "Comparing Models on Lookback Horizons" subsection of the Results, the performance of various models is compared across different datasets and lookback horizons of 48 hours in Figure 6a, 168 hours in Figure 6b and 336 hours in Figure 6c. The models under consideration include Ridge Regression, Gradient Boosting Models (GBM), Long Short-Term Memory



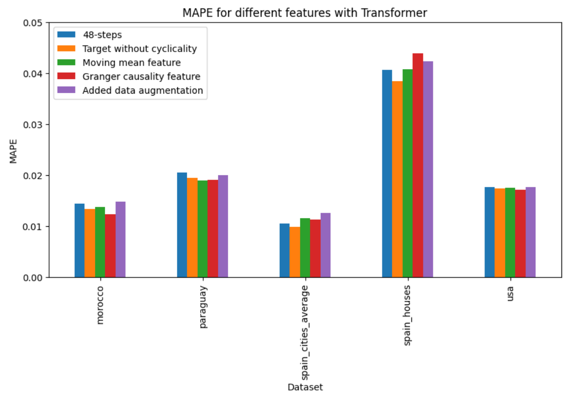
(a) Lookback horizon Transformer MAPE



(b) Hyperparameter tuning Transformer



(c) Weather feature for Transformer. The bars for spain houses goes to 0.054 for the baseline and 0.091 for the model with temperature and time encoding.



(d) Other features with Transformer

Figure 5: Transformer model evaluation

(LSTM), Bidirectional LSTM (BiLSTM), and Transformer.

The analysis reveals several patterns in the performance of these models across the datasets and lookback horizons. Ridge Regression consistently outperforms the other models significantly on the Morocco dataset, indicating its superior performance in this particular context.

Similarly, the GBM model demonstrates remarkable performance on the Spain Cities dataset, consistently outperforming the other models. This result highlights the effectiveness of the GBM model in capturing the underlying patterns within the Spain Cities dataset.

Conversely, the BiLSTM model exhibits the poorest performance across all datasets and lookback horizons, closely followed by the LSTM model as the second-worst performer. This finding suggests that these two recurrent neural network-based models may not be well-suited for the short-term electricity consumption forecasting task considered in this study.

The Transformer model demonstrates a more competitive performance, consistently ranking as either the best or second-best model across all datasets, except for the USA dataset, where it ranks second and third best. This result emphasizes the overall strong performance of the Transformer model in predicting electricity consumption patterns across various contexts.

In summary, this comparison of models on lookback horizons reveals distinct patterns in their performance across different datasets. Ridge Regression excels in the Morocco dataset, GBM dominates the Spain Cities dataset, and the Transformer model consistently performs well in most cases. On the other hand, the BiLSTM and LSTM models lag behind, indicating potential limitations in their applicability to short-term electricity consumption forecasting.

4.8 Production Settings

In the "Production Setting" section of the Results, the performance of various models is evaluated across different development timeframes, specifically 2-hour in Figure 7a, 6-hour in Figure 7b, 24-hour in Figure 7c, and 48-hour production settings in Figure 7d. The models under consideration include Transformer, Ridge Regression, Gradient Boosting Models (GBM), Long Short-Term Memory (LSTM), and Bidirectional LSTM (BiLSTM).

The analysis reveals that the Transformer model's performance varies across different production settings. At the 2-hour production setting, the Transformer model is slightly worse than Ridge but gradually becomes the best performer, significantly outperforming the other models in subsequent settings.

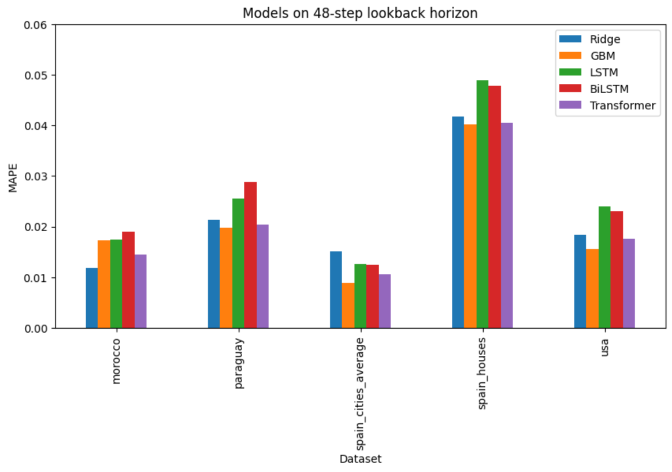
For the Paraguay dataset, the Transformer model consistently performs the best across all production settings, followed by Ridge and GBM as the second-best performers, while LSTM and BiLSTM models demonstrate the poorest performance.

In the Spain Cities dataset, Ridge consistently underperforms, with LSTM and BiLSTM models closely following. The GBM and Transformer models alternate in achieving the best performance in this dataset.

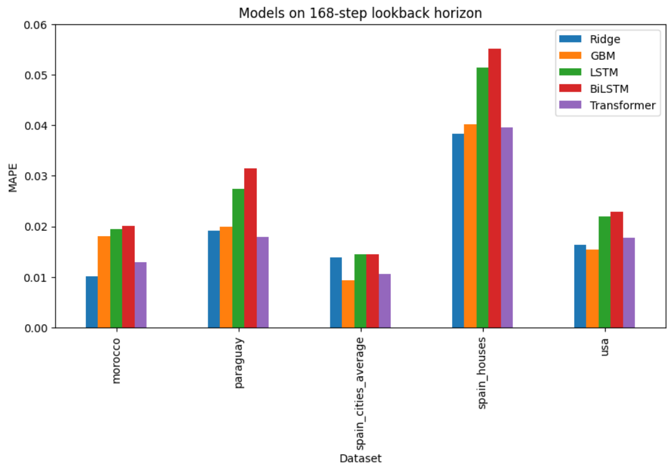
Regarding the Spain Houses dataset, the Transformer model initially outperforms the other models but is eventually overtaken by LSTM and BiLSTM models, which exhibit superior performance at the 48-hour production setting.

In the case of the USA dataset, the Transformer model maintains a significant lead in all production settings, except for the 6-hour setting, where it is narrowly outperformed by the GBM model.

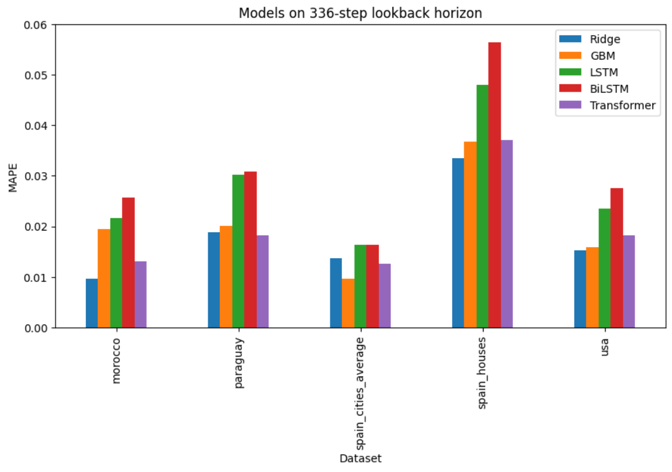
To summarize, the Transformer model demonstrates varying performance across different production settings and datasets. At the 2-hour production setting, it is significantly the best performer for three datasets and equally the best for the remaining two. By the 6-hour setting, the Transformer model is significantly the best for two datasets and equally the best for the other three. In the 24-hour setting, the Transformer model outperforms the other models in three datasets but is outperformed in the other two. Finally, at the 48-hour setting, the Transformer model once again emerges as the best performer in three datasets, while being outperformed in the remaining two.



(a) MAPE Comparison for 48 hour lookback

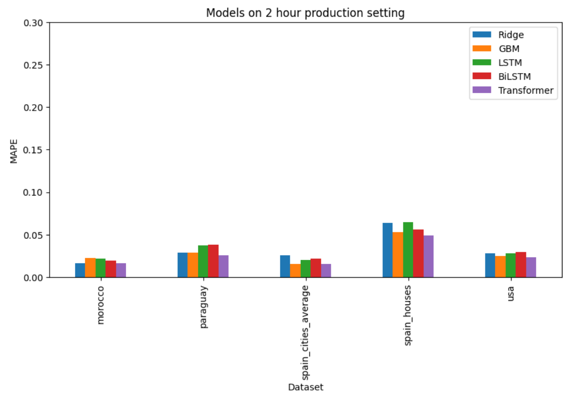


(b) MAPE Comparison for 168 hour lookback

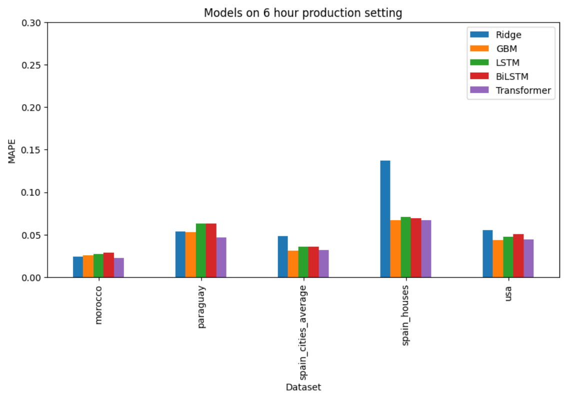


(c) MAPE Comparison for 336 hour lookback

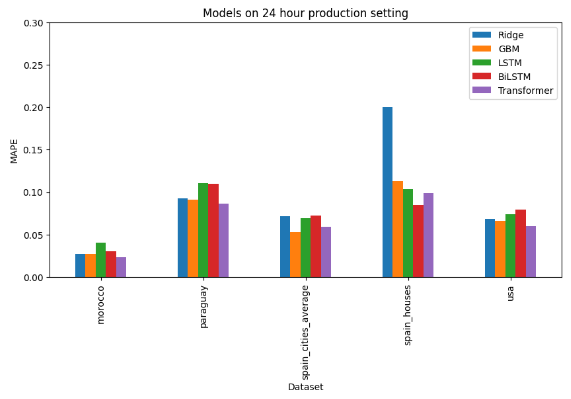
Figure 6: MAPE Comparisons for different lookback periods



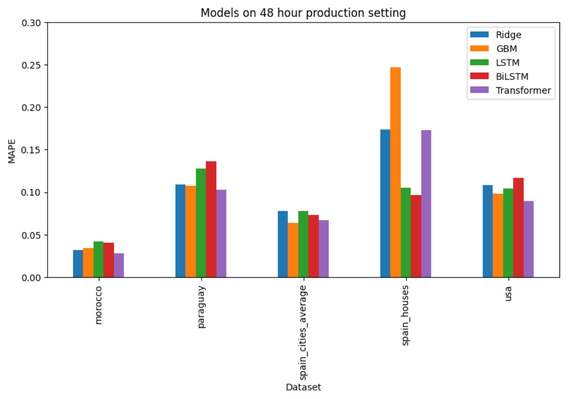
(a) Production settings for 2 hours forward in time



(b) Production settings for 6 hours forward in time



(c) Production settings for 24 hours forward in time



(d) Production settings for 48 hours forward in time

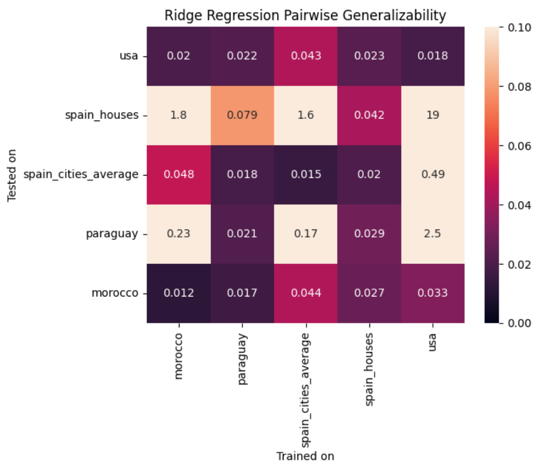
Figure 7: Production settings for different hours forward in time

4.9 Pairwise generalizability

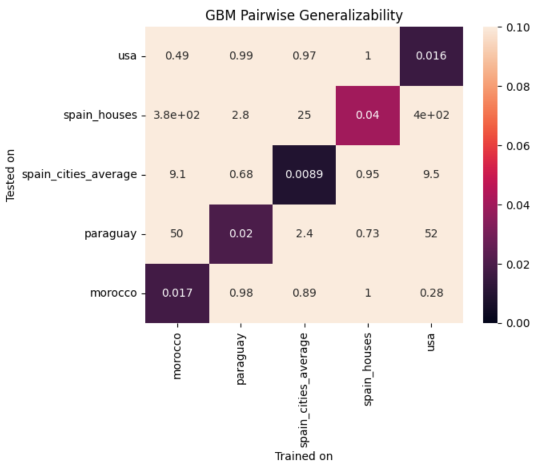
In the "Pairwise Generalizability" section of the Results, the ability of the various models to generalize across different datasets is assessed. The Gradient Boosting Models (GBM) model exhibited poor generalization to other datasets as shown in Figure 8b, indicating potential limitations in its applicability across diverse electricity consumption forecasting scenarios.

In contrast, the Transformer model in Figure 8e and Ridge Regression model in Figure 8a demonstrated varying degrees of success in generalizing to different datasets. When taking all of these comparisons in and finding which model performed best on each pairwise comparison, we end up with the table shown in Table 3. Specifically, the Transformer model outperformed the other models in 11 of the pairwise tests, while the Ridge Regression model displayed superior generalization in the remaining 9 tests. These results show that both the Transformer and Ridge Regression models possess a higher capacity to generalize across different datasets.

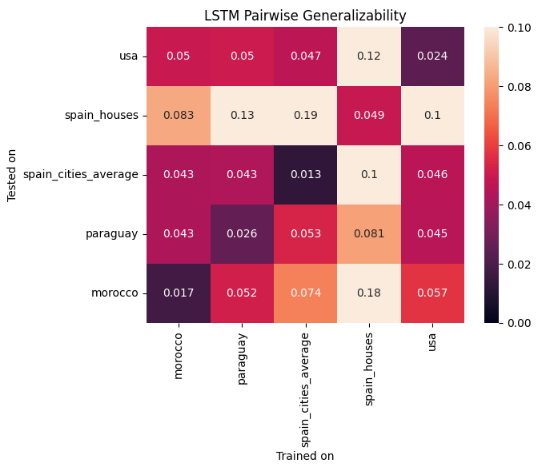
The LSTM model shown in Figure 8c and the BiLSTM model shown in Figure 8d performed worse than the transformer on every pairwise comparison.



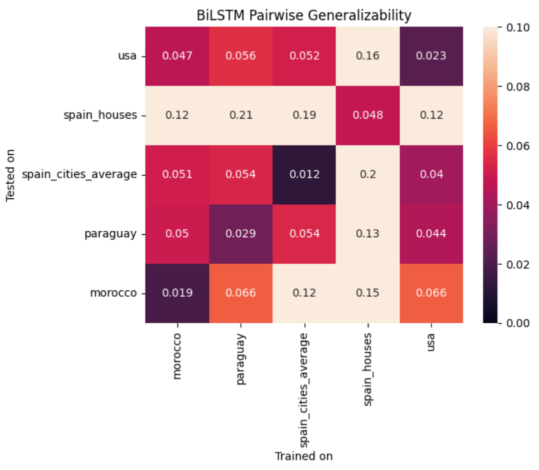
(a) Pairwise generalizability for Ridge.



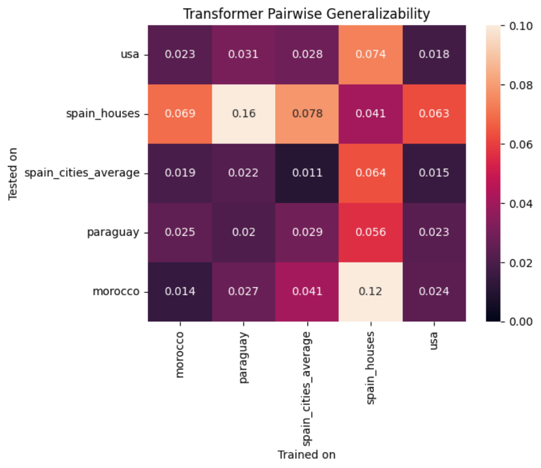
(b) Pairwise generalizability for GBM.



(c) Pairwise generalizability for LSTM.



(d) Pairwise generalizability for BiLSTM.



(e) Pairwise generalizability for Transformer.

Figure 8: Pairwise generalizability for different models

Tested on / Trained on	Morocco	Paraguay	Spain Cities	Spain Houses	USA
USA	Ridge	Ridge	Transformer	Ridge	-
Spain Houses	Transformer	Ridge	Transformer	-	Transformer
Spain Cities	Transformer	Ridge	-	Ridge	Transformer
Paraguay	Transformer	-	Transformer	Ridge	Transformer
Morocco	-	Ridge	Transformer	Ridge	Transformer

Table 3: View of which model generalized best for each pairwise testing.

4.10 Leave-one-out generalizability

In the Leave-one-out study shown in Figure 9, the performance of various models is trained on all but one dataset, then evaluated on the left out dataset. There is only one clear winner here, which is the Ridge model. It beat every other model on all the datasets.

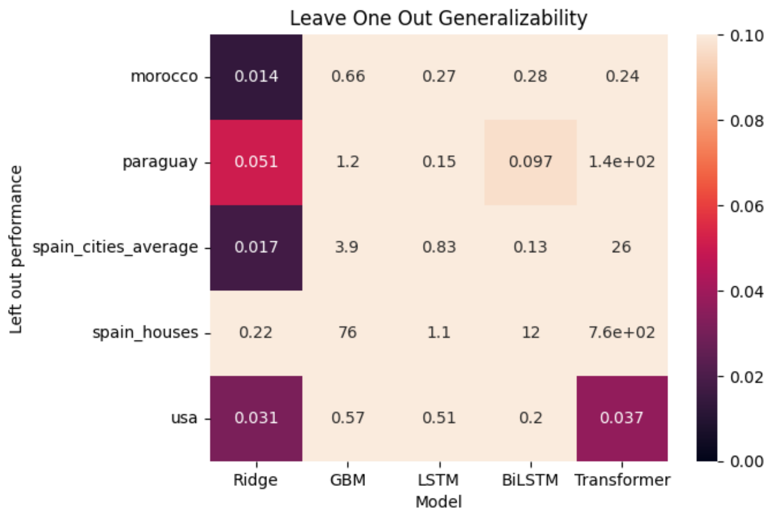


Figure 9: Leave one out generalizability.

4.11 Single Prediction Explainability

Figure 10 provides an example of how the Local Interpretable Model-agnostic Explanations (LIME) method can be employed for single prediction explainability. This specific prediction is derived from the Seville dataset recorded on 2018-10-03 at 09:00:00. The predicted value was less than 1% different from the actual value, reflecting the accuracy of the prediction.

To interpret this figure, let's focus on its three main components. The leftmost part represents the predicted value. In the middle section, the feature contributions are shown, indicating how much each feature helped increase or decrease the expected consumption for the given timestamp. These feature contributions are represented as horizontal bars; a orange bar demonstrates that a feature increases the predicted consumption, while a blue bar means the feature lowers it. The length of these bars indicates the magnitude of their contribution.

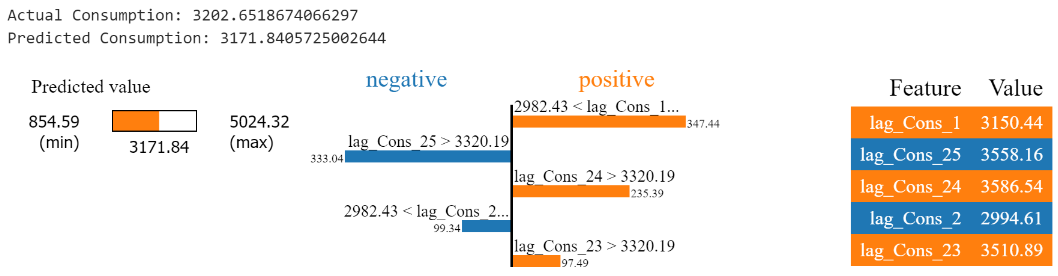
The rightmost section of the figure provides the feature values used to make the prediction.

The features are color-coded to denote whether the specific value helped raise (orange) or lower (blue) the predicted consumption.

The way LIME works is by generating 'neighborhood' data through random perturbations of features from the instance. It then learns locally weighted linear models on this neighborhood data to explain the predictions in an interpretable way.

This form of single prediction explainability is especially useful when we need to understand the contribution of different features to a specific prediction. In the context of this research, it helps determine how different factors influence the expected energy consumption at a given time.

In terms of the content shown in the figure, it meets explainability needs by clearly demonstrating how each feature contributes to the final prediction and providing a visual representation of the influence of each feature. This approach of using LIME and its visual outputs allows stakeholders to understand the model's decision-making process at a granular level, enhancing trust and facilitating better decision-making.



```
Actual Consumption: 3202.6518674066297
Predicted Consumption: 3171.8405725002644
```

Feature	Value
lag_Cons_1	3150.44
lag_Cons_25	3558.16
lag_Cons_24	3586.54
lag_Cons_2	2994.61
lag_Cons_23	3510.89

Figure 10: Example of single prediction explainability for Transformer model

4.12 model explainability

In Table 4 the model explainability of the Ridge regression is shown with regards to its feature importance for the different datasets, and showing the top five heavily weighted features. Testing the feature importance for the other features from Figure 1d did not change the features or orders of the initial model explainability table, but did slightly decrease the magnitude of the feature importances, which is to be expected when adding more features.

When doing the same with the GBM we end up with the table shown in Table 5. There are some differences in both the features being seen as most important as well as the magnitude to the most important feature. Similarly as with the Ridge model, testing the feature importance for the other features from Figure 2d did not change the features or orders of the initial model explainability table, but did slightly decrease the magnitude of the feature importances. The only exception to this was the GBM model with data augmentation shown in Table 6, where the most important features have a different order.

Morocco		Paraguay		Spain Cities		Spain Houses		USA	
Feature	Weight	Feature	Weight	Feature	Weight	Feature	Weight	Feature	Weight
Lag_1	2.43	Lag_1	2.03	Lag_1	3.62	Lag_1	3.26	Lag_1	4.78
Lag_25	2.05	Lag_25	1.00	Lag_2	0.39	Lag_25	0.28	Lag_25	1.44
Lag_24	1.62	Lag_24	0.94	Lag_25	0.33	Lag_24	0.11	Lag_2	0.90
Lag_2	0.09	Lag_23	0.03	Lag_24	0.19	Lag_2	0.04	Lag_24	0.51
Lag_26	0.06	Lag_26	0.03	Lag_23	0.04	Lag_14	0.03	Lag_26	0.14

Table 4: Ridge 48 step autocorrelation feature importances

Morocco		Paraguay		Spain Cities		Spain Houses		USA	
Feature	Weight	Feature	Weight	Feature	Weight	Feature	Weight	Feature	Weight
Lag_24	1.15	Lag_1	1.84	Lag_1	1.95	Lag_1	1.13	Lag_1	1.91
Lag_1	0.27	Lag_24	0.06	Lag_2	0.03	Lag_2	0.03	Lag_2	0.02
Lag_25	0.03	Lag_25	0.04	Lag_23	0.01	Lag_3	0.01	Lag_25	0.02
Lag_26	0.01	Lag_23	0.03	Lag_21	0.01	Lag_4	0.00	Lag_48	0.01
Lag_48	0.01	Lag_26	0.02	Lag_24	0.01	Lag_5	0.00	Lag_3	0.01

Table 5: GBM 48 step autocorrelation feature importances

Morocco		Paraguay		Spain Cities		Spain Houses		USA	
Feature	Weight	Feature	Weight	Feature	Weight	Feature	Weight	Feature	Weight
Lag_24	0.65	Lag_1	1.28	Lag_1	1.03	Lag_1	1.26	Lag_1	1.46
Lag_1	0.20	Lag_24	0.06	Lag_2	0.10	Lag_2	0.06	Lag_2	0.02
Lag_48	0.02	Lag_2	0.03	Lag_23	0.02	Lag_7	0.00	Lag_3	0.01
Lag_23	0.01	Lag_26	0.03	Lag_24	0.01	Lag_5	0.00	Lag_26	0.01
Lag_2	0.01	Lag_23	0.03	Lag_28	0.01	Lag_4	0.00	Lag_22	0.01

Table 6: GBMfeature importances with data augmentation

5 Discussion

The results from this study offer insightful conclusions regarding the performance and applicability of different models in predicting electricity consumption patterns. The findings are distilled and elaborated on to understand their implications.

5.1 Interpretation of results

The primary objective of this research was to compare the performance, generalizability, and explainability of transformer-based models with both simple machine learning methods, such as Ridge Regression and Gradient Boosting Models (GBM), and more advanced deep learning methods, including Long Short-Term Memory (LSTM) and Bidirectional LSTM (BiLSTM) networks.

The Ridge Regression model demonstrated robust performance on certain datasets, particularly the Morocco dataset. This robustness is likely attributed to Ridge Regression's aptitude for dealing with multicollinearity, a prevalent issue in time series forecasting. However, the model did not consistently perform well across all datasets, suggesting its success is more context-dependent rather than generalizable. This was unexpected given the simplicity of Ridge Regression compared to deep learning models. This finding challenges the common assumption that more complex models necessarily yield better performance, and suggests that simpler models may be more suitable for certain datasets, possibly due to their ability to avoid overfitting.

GBM, on the other hand, excelled remarkably on the Spain Cities dataset. While GBM may be adept at capturing the underlying patterns within this dataset, it may not necessarily extrapolate well to other datasets. This could potentially influence the choice of model in future energy consumption forecasting tasks, favoring Transformer models over LSTM and BiLSTM models.

LSTM and BiLSTM models did not perform optimally in the single-step ahead forecasting task. This was surprising given the renowned capabilities of these models in handling time-series data. This finding suggests that LSTM and BiLSTM models may have limitations in their ability to handle certain forecasting tasks, possibly due to issues such as vanishing or exploding gradients, high dimensionality, and potential noise in the data. This could have significant implications for the choice of models in future forecasting tasks, suggesting that despite their popularity, LSTM and BiLSTM models may not always be the best choice.

The Transformer model consistently outperformed LSTM and BiLSTM models across various datasets and lookback horizons. The superior performance of the Transformer model, which is equipped with a self-attention mechanism and the ability to capture long-term dependencies in the data, suggests that it is an effective tool for time-series forecasting tasks such as electricity consumption forecasting. This aligns with existing literature that highlights the strengths of the Transformer model, and provides a datapoint supporting its effectiveness in predicting electricity consumption.

The results also revealed several interesting patterns and relationships. For example, the performance of the Transformer model consistently improved with the inclusion of adding the target without cyclicality as a feature. This suggests that feature engineering, which involves creating new input features from the existing data, can slightly improve the performance of machine learning models in energy consumption forecasting for these five datasets. However, the effects of incorporating additional features like lagged temperature and time encodings varied across the models. For Ridge Regression, these features offered little to no improvement. Conversely, they led to minor improvements for the GBM model but deteriorated LSTM and BiLSTM model's performance. Transformer model exhibited the most consistent improvement with these features.

The results also showed that the inclusion of temperature just sometimes gave a slight performance increase. Most times adding lagged temperatures as features gave no change, or even slightly worse predictive power. This comes in contrast to the expectation that including data about temperatures would have a significant impact on the electricity consumption.

In conclusion, the results of this study not only provide a comparison of the performance of various models, but also offer valuable insights into the strengths and weaknesses of these models, and provide a foundation for future research in this area. The differences between these findings and previous research highlight the importance of considering a range of models and settings in energy consumption forecasting, and suggest potential directions for future research.

Model training speed was another factor considered in this study. Ridge models required roughly one second, while GBM models took approximately three minutes. In contrast, LSTM models required about three hours due to their CPU-dependent nature. The Transformer models were faster, at around one hour, owing to their GPU-dependent training process. This could have implications for the practical application of these models, especially in scenarios where real-time forecasting is required.

Hyperparameter tuning also played a significant role in model performance. Ridge Regression model's performance demonstrated robustness to changes in regularization strength, making it less sensitive to hyperparameters. Meanwhile, GBM model's performance improved with an increase in the number of estimators and leaves, although this also raised computational complexity. Fine-tuning of hyperparameters for LSTM and BiLSTM models was essential to prevent exploding gradients. This suggests that careful consideration and adjustment of model hyperparameters is crucial for optimal performance.

Model explainability, as demonstrated for the Transformer model using the LIME method, can help understand the contribution of different features to a prediction. This knowledge could be valuable in identifying key drivers of electricity consumption patterns, further informing interventions and policies to manage electricity demand. But both the Ridge and GBM have the easiest to understand model explainability.

The Ridge Regression and Transformer models demonstrated superior performance in generalizing to different datasets, indicating their potential as versatile forecasting tools. Conversely, GBM, LSTM, and BiLSTM models exhibited limitations in transferability across diverse electricity consumption forecasting scenarios. This highlights the importance of model generalizability and transferability in forecasting tasks, and suggests that these factors should be considered when selecting models for future research and practice.

5.2 Implications

The results of this study have several broader implications for the field of energy consumption forecasting and suggest potential directions for future research.

Firstly, the findings suggest that incorporating additional features such as lagged temperature and time encodings could significantly improve forecasting accuracy. This aligns with the broader trend in machine learning towards feature engineering, which involves creating new input features from the existing data to enhance model performance. This could have practical applications in the development of more accurate and reliable forecasting models, which could in turn support more effective energy management and policy decisions.

Secondly, the superior performance of the Transformer model across various datasets and lookback horizons highlights its potential for energy consumption forecasting. Given its computational efficiency compared to LSTM networks and ability to capture long-term dependencies in the data, Transformer models could be a valuable tool for researchers and practitioners in the field. This could influence future research directions, with more studies investigating the potential of Transformer models in energy forecasting and other related fields.

The robust performance of the Ridge Regression model on certain datasets also has important implications. This finding challenges the common assumption that more complex models necessarily yield better performance and suggests that simpler models may be more suitable for certain datasets. It is also worth noting that none of the papers in the literature review had ridge regression as a benchmark to beat. This could influence the choice of models in future research and practice, with a greater emphasis on the potential benefits of simpler models, which could perform as well or better than deep neural networks on some datasets, while providing better generalizability and explainability. It could also inform policy decisions, by highlighting the importance of considering a range of models, including simpler ones, in energy consumption forecasting.

The suboptimal performance of LSTM and BiLSTM models in the single-step ahead forecasting task suggests potential limitations of these models in handling certain forecasting tasks. This finding could influence future research directions, with more studies investigating the limitations of LSTM and BiLSTM models and exploring potential ways to overcome these limitations. It could also have practical implications, by informing the choice of models in energy consumption forecasting tasks.

The findings from this study extend the current understanding of the performance and limitations of various forecasting models in different contexts. In particular, the superiority of Transformer models over LSTM models in several aspects. Future steps could be to compare Transformer models to other model architectures that have shown improvement over the LSTM benchmark, such as CNN-LSTM or TCN.

However, further research is needed to understand why Ridge Regression performs exceptionally well, despite not being tested in any of the papers from the literature review. Despite Ridge regressions good performance here, the established benchmark is most often set to be either ARIMA or a vanilla LSTM networks, so these results shed new light on the forecasting capabilities of Ridge regression, and the need to include it as a benchmark models for future comparisons.

The study's implications extend to the practical realm as well. The choice of model for electricity consumption forecasting should consider the specific dataset, lookback horizon, production setting, and additional features incorporated. Additionally, the differences in model training speeds have practical implications regarding the choice of models in time-constrained scenarios.

The relative simplicity of the datasets used in this study might favour shallow models with fewer parameters, such as Ridge Regression and GBM, over deep learning models including LSTM, BiLSTM, and Transformer. When dealing with datasets exhibiting more complex structures or higher dimensionalities, deep learning models might demonstrate superior performance. This suggests that the nature of the dataset should be a key consideration in the choice of forecasting model.

In conclusion, this study contributes to the field of energy consumption forecasting by providing a comprehensive evaluation of various machine learning and deep learning models. The findings offer valuable insights into the strengths and weaknesses of these models, and suggest potentialdirections for future research and practice. They could also influence policy decisions, by informing the choice of models in energy consumption forecasting and highlighting the potential benefits of feature engineering and simpler models.

The findings also underscore the importance of model explainability in energy consumption forecasting. As demonstrated for the Transformer model using the LIME method, understanding the contribution of different features to a prediction can be valuable in identifying key drivers of electricity consumption patterns. This could further inform interventions and policies to manage electricity demand. However, it's worth noting that both the Ridge and GBM models have easier to understand model explainability, which could be a significant advantage in certain applications where interpretability is crucial.

Finally, the study's findings highlight the importance of model generalizability and transferability in forecasting tasks. The Ridge Regression and Transformer models demonstrated superior performance in generalizing to different datasets, indicating their potential as versatile forecasting tools. Conversely, GBM, LSTM, and BiLSTM models exhibited limitations in transferability across diverse electricity consumption forecasting scenarios. This suggests that these factors should be considered when selecting models for future research and practice, and underscores the need for further research to improve the generalizability and transferability of these models.

5.3 Limitations

While this study provides valuable insights into the performance, generalizability, and explainability of various machine learning and deep learning models for forecasting electricity consumption, it is important to acknowledge its limitations, as they might have influenced the results.

Firstly, the Ridge Regression model, although robust across several lookback horizons for a specific dataset, showed inconsistent performance across different datasets. This suggests its success might be context-dependent, and its performance might not be as strong under different conditions or with different datasets. Similarly, LSTM and BiLSTM models did not perform optimally in this study, but they might perform better under different conditions or with different datasets. These considerations underscore the importance of taking into account the specific conditions and limitations of a study when interpreting and applying its findings.

Secondly, the performance of the models was evaluated based on specific datasets and under certain parameter settings. Different datasets or parameter settings might yield different results, potentially influencing the conclusions drawn from this study. Moreover, the study focused on specific models, namely Ridge Regression, Gradient Boosting Model, LSTM, BiLSTM, and Transformer models. While these models represent a range of simple to advanced machine learning and deep learning methods, there are many other models that were not considered in this study. The performance of these other models might be different from the models evaluated in this study, potentially offering different perspectives on the task of energy consumption forecasting.

Thirdly, while the study aimed to assess the explainability of the models, explainability in machine learning is a complex and multifaceted concept. The methods used to assess explainability in this study might not capture all aspects of explainability, potentially influencing the interpretation of the results. For instance, the study focused on the interpretability of the model's predictions, but other aspects of explainability, such as the transparency of the model's decision-making process or the comprehensibility of the model's structure, were not thoroughly assessed. These aspects are only easily accessible for the classical machine learning models. If these aspects were considered, the conclusions on the explainability of the models might be different.

Lastly, in the data augmentation process, a 2% noise was added to the datasets. However, a caveat arises that GBM's mechanism of binning data could potentially lead to data leakage in this scenario. This could be falsely inflating the model's performance, emphasising the importance of careful data augmentation strategies when using models like GBM.

5.4 Future Work

The results of this study, while providing valuable insights into the performance of various machine learning and deep learning models in energy consumption forecasting, also highlight several areas for future exploration:

- **Increasing the number of datasets:** One of the limitations for selection of datasets was that they contained temperature data. But with the discovery that temperature did not provide a performance increase in most cases, this requirement could be relaxed such that the different models could be compared on a much larger number of datasets.
- **Exploring Other Models:** This study focused on a specific set of models, namely Ridge Regression, Gradient Boosting Model, LSTM, BiLSTM, and Transformer models. However, there are numerous other models that could be explored. Future research

could investigate the performance of these other models in energy consumption forecasting. For instance, ARIMA, a widely used model in time series forecasting, and hybrid models such as CNN-LSTM or TCN could be considered. This could potentially uncover new models that perform well in this task, expanding the toolkit of effective models for researchers and practitioners in the field.

- **Investigating Performance Across Diverse Conditions:** The performance of the models in this study was evaluated under specific conditions, using particular datasets and parameter settings. However, the performance of these models might vary under different conditions. Future research could therefore explore the robustness of these models across a variety of datasets and parameter settings. This could help to establish a more comprehensive understanding of the models' performance and their generalizability.
- **Deepening Understanding of Model Explainability:** The concept of explainability in machine learning is complex and multifaceted. While this study assessed the explainability of the models based on the interpretability of their predictions, other aspects of explainability were not thoroughly assessed. Future research could delve deeper into these aspects, exploring methods for assessing the transparency of the model's decision-making process or the comprehensibility of the model's structure. This could lead to a more nuanced understanding of explainability in machine learning and deep learning models, which could inform the development of models that are not only high-performing, but also transparent and understandable.
- **Investigating the Role of Feature Engineering:** The results showed that the performance of machine learning models in energy consumption forecasting can be improved by incorporating additional features such as lagged temperature and time encodings. Future research could further explore the role of feature engineering in improving the performance of forecasting models.

In conclusion, while this study contributes to the field of energy consumption forecasting by providing a comprehensive evaluation of various models, it also opens up several avenues for future research. These future directions not only build on the findings of this study, but also have significant implications for the future direction of research in this field.

5.5 Conclusion

This study embarked on a comprehensive evaluation of various machine learning and deep learning models for forecasting electricity consumption. The models evaluated included Ridge Regression, Gradient Boosting Model, LSTM, BiLSTM, and Transformer models. The performance, generalizability, and explainability of these models were assessed using multiple datasets and under various conditions.

The results of the study revealed several intriguing insights. The Transformer model consistently outperformed the other models across various datasets and lookback horizons, highlighting its potential as a powerful tool for energy consumption forecasting. The Ridge Regression model also demonstrated robust performance on certain datasets, challenging the common assumption that more complex models necessarily yield better performance. On the other hand, LSTM and BiLSTM models did not perform optimally in the single-step ahead forecasting task, suggesting potential limitations of these models in handling certain forecasting tasks.

The study also underscored the importance of feature engineering in improving model performance. The inclusion of additional features such as lagged temperature and time encodings was found to enhance the performance of the models, particularly the Transformer model. However, the effects of these additional features varied across the models, indicating that the benefits of feature engineering might be model-specific.

The findings of this study contribute to the ongoing discussions in the field of energy consumption forecasting. By comparing the performance of various models, this study offers some insights into their potential strengths and limitations. While these observations may not be definitive, they could serve as a starting point for further research and could potentially inform model selection in future studies.

However, the study also had several limitations, including the specific conditions under which the models were evaluated and the limited range of models considered. These limitations highlight the need for further research to explore the performance of other models under diverse conditions, and to delve deeper into the concept of explainability in machine learning.

Looking forward, several avenues for future research were identified. These include exploring other models, investigating model performance across diverse conditions, deepening the understanding of model explainability, addressing the trade-off between model performance and explainability, and further investigating the role of feature engineering. These future directions not only build on the findings of this study, but also have significant implications for the future direction of research in this field.

In conclusion, this study contributes to the field of energy consumption forecasting by providing a comprehensive evaluation of various models and by highlighting several avenues for future research. The findings of the study offer valuable insights for researchers and practitioners in the field, and pave the way for further advancements in energy consumption forecasting.

References

[1] T. Hong, P. Pinson, Y. Wang, R. Weron, D. Yang, and H. Zareipour, "Energy forecasting: A review and outlook," Department of Operations Research and Business Intelligence, Wroclaw University of Science and Technology, WORking papers in Management Science (WORMS) WORMS/20/08, May 2020. [Online]. Available: https://ideas.repec.org/p/ahh/wpaper/worms2008.html

[2] R. Patel, M. Patel, and N. Patel, "Electrical load forecasting using machine learning methods, rnn and lstm," *Xi'an Dianzi Keji Daxue Xuebao/Journal of Xidian University*, vol. 14, pp. 1376 – 1386, 04 2020.

[3] S. Kumar, L. Hussain, S. Banarjee, and M. Reza, "Energy load forecasting using deep learning approach-lstm and gru in spark cluster," in *2018 Fifth International Conference on Emerging Applications of Information Technology (EAIT)*, 2018, pp. 1–4.

[4] L. Guo, L. Wang, and H. Chen, "Electrical load forecasting based on lstm neural networks," in *Proceedings of the 2019 International Conference on Big Data, Electronics and Communication Engineering (BDECE 2019).* Atlantis Press, 2019, pp. 107–111. [Online]. Available: https://doi.org/10.2991/acsr.k.191223.024

[5] M. R. Islam, A. Al Mamun, M. Sohel, M. L. Hossain, and M. M. Uddin, "Lstm-based electrical load forecasting for chattogram city of bangladesh," in *2020 International Conference on Emerging Smart Computing and Informatics (ESCI)*, 2020, pp. 188–192.

[6] A. A. Mamun, M. Hoq, E. Hossain, and R. Bayindir, "A hybrid deep learning model with evolutionary algorithm for short-term load forecasting," in *2019 8th International Conference on Renewable Energy Research and Applications (ICRERA)*, 2019, pp. 886–891.

[7] J. F. Torres, F. Martínez-Álvarez, and A. Troncoso, "A deep lstm network for the spanish electricity consumption forecasting," *Neural Computing and Applications*, vol. 34, no. 13, pp. 10 533–10 545, Jul 2022. [Online]. Available: https://doi.org/10.1007/s00521-021-06773-2

[8] Lum, Kai Lok, Mun, Hou Kit, Phang, Swee King, and Tan, Wei Qiang, "Industrial electrical energy consumption forecasting by using temporal convolutional neural networks," *MATEC Web Conf.*, vol. 335, p. 02003, 2021. [Online]. Available: https://doi.org/10.1051/matecconf/202133502003

[9] V. Gundu and S. P. Simon, "Quality analysis of combined similar day and day ahead short-term load forecasting using recurrent neural networks," *Sādhanā*, vol. 47, no. 1, p. 43, Feb 2022. [Online]. Available: https://doi.org/10.1007/s12046-022-01825-2

[10] X. Shao, C.-S. Kim, and P. Sontakke, "Accurate deep model for electricity consumption forecasting using multi-channel and multi-scale feature fusion cnn–lstm," *Energies*, vol. 13, no. 8, 2020. [Online]. Available: https://www.mdpi.com/1996-1073/13/8/1881

[11] L. Wu, C. Kong, X. Hao, and W. Chen, “A short-term load forecasting method based on gru-cnn hybrid neural network model,” *Mathematical Problems in Engineering*, vol. 2020, p. 1428104, Mar 2020. [Online]. Available: https://doi.org/10.1155/2020/1428104

[12] H. Kuang, Q. Guo, S. Li, and H. Zhong, “Short-term power load forecasting method in rural areas based on cnn-lstm,” in *2021 IEEE 4th International Electrical and Energy Conference (CIEEC)*, 2021, pp. 1–5.

[13] H. Eskandari, M. Imani, and M. P. Moghaddam, “Convolutional and recurrent neural network based model for short-term load forecasting,” *Electric Power Systems Research*, vol. 195, p. 107173, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378779621001541

[14] G. Dudek, P. Pełka, and S. Smyl, “3ets+rd-lstm: A new hybrid model for electrical energy consumption forecasting,” in *Neural Information Processing*, H. Yang, K. Pasupa, A. C.-S. Leung, J. T. Kwok, J. H. Chan, and I. King, Eds. Cham: Springer International Publishing, 2020, pp. 519–531.

[15] N. Pooniwala and R. Sutar, “Forecasting short-term electric load with a hybrid of arima model and lstm network,” in *2021 International Conference on Computer Communication and Informatics (ICCCI)*, 2021, pp. 1–6.

[16] T. Bashir, C. Haoyong, M. F. Tahir, and Z. Liqiang, “Short term electricity load forecasting using hybrid prophet-lstm model optimized by bpnn,” *Energy Reports*, vol. 8, pp. 1678–1686, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352484721015067

[17] A. A. Muzumdar, C. N. Modi, M. G. M, and C. Vyjayanthi, “Designing a robust and accurate model for consumer-centric short-term load forecasting in microgrid environment,” *IEEE Systems Journal*, vol. 16, no. 2, pp. 2448–2459, 2022.

[18] K. Ke, S. Hongbin, Z. Chengkang, and C. Brown, “Short-term electrical load forecasting method based on stacked auto-encoding and gru neural network,” *Evolutionary Intelligence*, vol. 12, no. 3, pp. 385–394, Sep 2019. [Online]. Available: https://doi.org/10.1007/s12065-018-00196-0

[19] Y. TAO, F. KONG, W. JU, H. LI, and R. HOU, “Cross-domain energy consumption prediction via ed-lstm networks,” *IEICE Transactions on Information and Systems*, vol. E104.D, no. 8, pp. 1204–1213, 2021.

[20] K. Sharma, Y. K. Dwivedi, and B. Metri, “Incorporating causality in energy consumption forecasting using deep neural networks,” *Annals of Operations Research*, Jul 2022. [Online]. Available: https://doi.org/10.1007/s10479-022-04857-3

[21] J. Guo, Y. Peng, Q. Zhou, and Q. Lv, “Enhanced lstm model for short-term load forecasting in smart grids,” in *Cloud Computing, Smart Grid and Innovative Frontiers in Telecommunications*, X. Zhang, G. Liu, M. Qiu, W. Xiang, and T. Huang, Eds. Cham: Springer International Publishing, 2020, pp. 650–662.

[22] L. H. Anh, G. H. Yu, D. T. Vu, J. S. Kim, J. I. Lee, J. C. Yoon, and J. Y. Kim, "Stride-tcn for energy consumption forecasting and its optimization," *Applied Sciences*, vol. 12, no. 19, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/19/9422

[23] S. Tsianikas, X. Xie, R. S. Puri, A. K. Parlikad, and D. W. Coit, "Comparison of neural network based approaches for short-term residential energy load forecasting," *IIE Annual Conference. Proceedings*, pp. 1–6, 2022, copyright - Copyright Institute of Industrial and Systems Engineers (IISE) 2022. [Online]. Available: https://www.proquest.com/scholarly-journals/comparison-neural-network-based-approaches-short/docview/2715838496/se-2?accountid=12870

[24] F. Cameron-Muller, D. Weeraddana, R. Chalapathy, and N. L. D. Khoa, "Dual-stage bayesian sequence to sequence embeddings for energy demand forecasting," in *Advances in Knowledge Discovery and Data Mining*, K. Karlapalem, H. Cheng, N. Ramakrishnan, R. K. Agrawal, P. K. Reddy, J. Srivastava, and T. Chakraborty, Eds. Cham: Springer International Publishing, 2021, pp. 277–289.

[25] N. J. Johannesen, M. L. Kolhe, and M. Goodwin, "Comparing recurrent neural networks using principal component analysis for electrical load predictions," in *2021 6th International Conference on Smart and Sustainable Technologies (SpliTech)*, 2021, pp. 1–6.

[26] H. Grigoryan, "Electricity consumption prediction using energy data, socio-economic and weather indicators. a case study of spain," in *2021 9th International Conference on Control, Mechatronics and Automation (ICCMA)*, 2021, pp. 158–164.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

[28] A. Zeyer, P. Bahar, K. Irie, R. Schlüter, and H. Ney, "A comparison of transformer and lstm encoder decoder models for asr," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 8–15.

[29] L. Pérez-Mayos, M. Ballesteros, and L. Wanner, "How much pretraining data do language models need to learn syntax?" 2021.

[30] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in neural information processing systems*, 2014, pp. 3320–3328.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[32] D. Han, Q. Liu, and W. Fan, "A new image classification method using cnn transfer learning and web data augmentation," *Expert Systems with Applications*, vol. 95, pp. 43–56, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417417307844

[33] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, pp. 206–215, 2019.

[34] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso, "Machine learning interpretability: A survey on methods and metrics," *Electronics*, vol. 8, no. 8, p. 832, 2019.

[35] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.

[36] A. Holzinger, "The next frontier: Ai we can really trust," in *Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, M. Kamp, I. Koprinska, A. Bibal, T. Bouadi, B. Frénay, L. Galárraga, J. Oramas, L. Adilova, Y. Krishnamurthy, B. Kang, C. Largeron, J. Lijffijt, T. Viard, P. Welke, M. Ruocco, E. Aune, C. Gallicchio, G. Schiele, F. Pernkopf, M. Blott, H. Fröning, G. Schindler, R. Guidotti, A. Monreale, S. Rinzivillo, P. Biecek, E. Ntoutsi, M. Pechenizkiy, B. Rosenhahn, C. Buckley, D. Cialfi, P. Lanillos, M. Ramstead, T. Verbelen, P. M. Ferreira, G. Andresini, D. Malerba, I. Medeiros, P. Fournier-Viger, M. S. Nawaz, S. Ventura, M. Sun, M. Zhou, V. Bitetta, I. Bordino, A. Ferretti, F. Gullo, G. Ponti, L. Severini, R. Ribeiro, J. Gama, R. Gavaldà, L. Cooper, N. Ghazaleh, J. Richiardi, D. Roqueiro, D. Saldana Miranda, K. Sechidis, and G. Graça, Eds. Cham: Springer International Publishing, 2021, pp. 427–440.

[37] S. Wachter, B. Mittelstadt, and L. Floridi, "Transparent, explainable, and accountable ai for robotics," *Science Robotics*, vol. 2, no. 6, p. eaan6080, 2017.

[38] S. Siami-Namini and A. S. Namin, "A comparison of arima and lstm in forecasting time series," *Proceedings of the 2019 7th International Conference on Behavioral, Economic, and Socio-Cultural Computing*, pp. 1–6, 2019.

[39] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The m4 competition: 100,000 time series and 61 forecasting methods," *International Journal of Forecasting*, vol. 36, no. 1, pp. 54–74, 2020.

[40] C. W. Granger, "Investigating causal relations by econometric models and cross-spectral methods," *Econometrica: Journal of the Econometric Society*, pp. 424–438, 1969.

[41] M. Ding, Y. Chen, and S. L. Bressler, "Granger causality: Basic theory and application to neuroscience," *Handbook of time series analysis*, vol. 1, pp. 437–460, 2006.

[42] A. B. Barrett, L. Barnett, and A. K. Seth, "Multivariate granger causality and generalized variance," *Phys. Rev. E*, vol. 81, p. 041907, Apr 2010. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.81.041907

[43] T. Hong and S. Fan, "Probabilistic electric load forecasting: A tutorial review," *International Journal of Forecasting*, vol. 32, no. 3, pp. 914–938, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169207015001508

[44] J. W. Taylor and P. E. McSharry, "Short-term load forecasting methods: An evaluation based on european data," *IEEE Transactions on Power Systems*, vol. 22, no. 4, pp. 2213–2219, 2007.

[45] M. De Felice, A. Alessandri, and P. M. Ruti, "Electricity demand forecasting over italy: Potential benefits using numerical weather prediction models," *Electric Power Systems Research*, vol. 104, pp. 71–79, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378779613001545

[46] A. Forootani, M. Rastegar, and A. Sami, "Short-term individual residential load forecasting using an enhanced machine learning-based approach based on a feature engineering framework: A comparative study with deep learning methods," *Electric Power Systems Research*, vol. 210, p. 108119, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378779622003431

[47] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970. [Online]. Available: http://www.jstor.org/stable/1267351

[48] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction.* Springer Science & Business Media, 2009.

[49] A. Y. Ng, "Feature selection, l1 vs. l2 regularization, and rotational invariance," in *Proceedings of the twenty-first international conference on Machine learning.* ACM, 2004, p. 78.

[50] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[51] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

[52] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[53] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning.* MIT Press, 2016, http://www.deeplearningbook.org.

[54] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[55] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[56] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[57] A. Graves, A.-r. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[58] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[59] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[60] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving language understanding by generative pre-training,” *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/language-unsupervised/language_understanding_paper. pdf*, 2018.

[61] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *arXiv preprint arXiv:1910.10683*, 2019.

[62] “Power consumption of tetouan city data set.” [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Power+consumption+of+Tetouan+city

[63] “Electric current consumption and meteorological data of alto parana, paraguay.” [Online]. Available: https://data.mendeley.com/datasets/hzfwzzsk8f/4

[64] “Hourly energy demand generation and weather.” [Online]. Available: https://www.kaggle.com/datasets/nicholasjhana/energy-consumption-generation-prices-and-weather

[65] “Energy consumption curves of 499 customers from spain.” [Online]. Available: https://fordatis.fraunhofer.de/handle/fordatis/215

[66] “Global energy forecasting competition 2012 - load forecasting.” [Online]. Available: https://www.kaggle.com/c/global-energy-forecasting-competition-2012-load-forecasting

[67] “Energy consumption and weather data collected continuously during a period of 29 months at challenger building (france).” [Online]. Available: https://zenodo.org/record/3445332#.Ybcj6n3MJ24

[68] “Load on the elia grid.” [Online]. Available: https://opendata.elia.be/explore/?refine.theme=Load&refine.publisher=Customers,%20Market%20and%20System&disjunctive.theme&disjunctive.dcat.granularity&disjunctive.dcat.accrualperiodicity&disjunctive.keyword&sort=explore.popularity_score

[69] “Smartmeter energy consumption data in london households.” [Online]. Available: https://data.london.gov.uk/dataset/smartmeter-energy-use-data-in-london-households

[70] “Hourly energy consumption.” [Online]. Available: https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption

[71] “Historical data (until december 2015).” [Online]. Available: https://www.entsoe.eu/data/data-portal/

[72] “Individual household electric power consumption data set.” [Online]. Available: https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption

[73] “Nyiso (new york independent system operator) hourly actual load.” [Online]. Available: http://www.energyonline.com/Data/GenericData.aspx?DataId=13&NYISO___Hourly_Actual_Load

[74] “Aggregated price and demand data.” [Online]. Available: https://aemo.com.au/en/energy-systems/electricity/national-electricity-market-nem/data-nem/aggregated-data

[75] “Commercial and residential hourly load profiles for all tmy3 locations in the united states.” [Online]. Available: https://data.openei.org/submissions/153?source=post_page---------------------------

[76] “Electricity data cnu.” [Online]. Available: https://github.com/andrewlee1807/tcns-with-nas/tree/main/Dataset/cnu-dataset