ELECTRICITY CONSUMPTION FORECAST OF RESIDENTIAL HOUSING USING DEEP NEURAL NETWORKS

by

Nicolai Bo Vanting

Spring / Summer – 2021

SPDS800 – Data Science Master's Thesis

University of Southern Denmark

Supervisors

Bo Nørregaard Jørgensen

Professor

Center for Energy Informatics

The Maersk Mc-Kinney Moller Institute

Richard Röttger

Associate Professor

Department of Mathematics and Computer Science

Acknowledgments

I would first like to thank Professor Bo Nørregaard Jørgensen for his supervision and support for this master’s thesis. Your feedback, input, and ideas pushed my work to a higher level.

I would also like to thank Associate Professor Richard Röttger for his technical feedback and supervision. Your knowledge and expertise in the field of deep learning were a fantastic help.

I would like to thank Associate Professor Zheng Ma for her support and supervision. Your feedback and methods were a huge help for the preparation of the literature review.

In addition, I would like to thank Center Denmark, Flexible Energy Denmark, and TREFOR for making the data used for this thesis available for research.

Finally, I want to thank my partner in life, Cæcilie, for always being there for me. Your cheerfulness and companionship always provide a happy distraction during my research.

Note to the reader

The literature review for this thesis has been reviewed and accepted for publication in the *OpenSpringer – Energy Informatics Journal* as a review paper and was presented at the *1st Energy Informatics.Academy Conference*, Asia, on the 29th of May 2021. The introduction also includes parts of this publication.

Abstract

I dette kandidatspeciale undersøges der, hvordan neurale netværk kan benyttes til at gøre forudsigelser af elforbrug fra private boliger i danske boligområder mere præcise, samt hvilke fordele eller ulemper der opstår ved brug af denne metode. Det øgede fokus på vedvarende energikilder i Danmark betyder, at forsyningsselskaber står overfor store udfordringer i elnettet med hensyn til nettets tilstand og kapacitet. Private boliger udgør en stor del af det samlede elforbrug, og med det stigende antal varmepumper og elbiler i disse områder, vil behovet for en højere elkapacitet kun blive større. Af denne grund er det afgørende, at forsyningsselskaberne kan forudsige elnettets tilstand på både kort og lang sigt. De seneste år har der været stor udvikling i brugen af neurale netværk til forudsigelser af elforbrug. Specialet er derfor baseret på en analyse og vurdering af state-of-the-art litteratur relateret til deep learning metoder, da denne litteratur er forudsætningen for at finde den rette arkitektur til det neurale netværk. Data er indsamlet i forbindelse med det nationale digitaliseringsprojekt, Flexible Energy Denmark, hvor forsyningsselskaber har gjort elmålerdata fra udvalgte testområder i Danmark tilgængelige for forskning. Det samlede elforbrug fra januar 2019 til april 2021 fra to boligområder i Kolding og Middelfart bruges som input sammen med vejrdata og tidsvariabler. Baseret på analysen af eksisterende litteratur anvendes en hybrid model bestående af to specialiserede neurale netværk til forudsigelsen af elforbruget for hvert område. Data bruges til at eksperimentere med, hvordan den hybride model præsterer sammenlignet med klassiske neurale netværk. Resultaterne viser, at det hybride neurale netværk har fordele og kan reducere afvigelsen på forudsigelsen med 40%, når der sammenlignes med de bedste benchmarkmodeller. På data fra 1. januar 2021 til 4. maj 2021 opnås der en gennemsnitlig numerisk afvigelse for forudsigelser for den næste time på henholdsvis 4.43kWh og 8.52kWh for det samlede elforbrug i boligområderne i Middelfart og Kolding. I Kolding området formåede den foreslåede model ikke at forbedre afvigelsen, når der sammenlignes med de bedste basismodeller. Der argumenteres for, at forskellen i præstation stammer fra en større andel installationer med varmepumper i boligområdet i Kolding. Dette angiver en svaghed i den foreslåede model og understreger behovet for at undersøge, hvordan ikke-menneskelig adfærd, såsom varmepumper, påvirker elforbruget og dermed forudsigelserne.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ANN – Artificial Neural Network
ARIMA – Autoregressive Integrated Moving Average
CNN – Convolutional Neural Network
DER – Distributed Energy Resources
DSO – Distribution System Operators
GRU – Gated Recurrent Unit
HVAC – Heating, Ventilation, and Air Conditioning
IoT – Internet of Things
LSTM – Long Short-Term Memory
MLP – Multi-Layer Perceptron
NARNN – Nonlinear Autoregressive Neural Network
PSO – Particle Swarm Optimization
ReLU – Rectified Linear Unit
RNN – Recurrent Neural Network
SCADA – Supervisory Control and Data Acquisition
SGD – Stochastic Gradient Descent

1. INTRODUCTION

An increasing number of distributed energy resources (DERs) introduces challenges to the energy system, e.g., grid balancing (Billanes, Ma, & Jørgensen, 2017; Ma, Værbak, Rasmussen, & Jøergensen, 2019). This challenge is especially crucial to distribution system operators (DSOs) who maintain control of the energy grid (Ma, Sommer, & Jørgensen, 2016). Compared to the large-scale battery, demand response is a better solution (Ma & Jørgensen, 2018). However, demand response might create enormous activities in the grid due to the communication between the electricity markets and the demand sides, which potentially affects the distribution grid stability (Christensen, Ma, & Jørgensen, 2021). Therefore, accurate predictions of the future loads in the distribution grid are necessary. Globally, households make up a large part of electricity consumption (Nejat, Jomehzadeh, Taheri, Gohari, & Abd. Majid, 2015). However, most distribution grid designs cannot afford a large number of DERs, especially synchronous charging (Ma, Christensen, & Jorgensen, 2021). Thus, it is critical for DSOs to accurately forecast energy load in the residential areas to improve their operation quality and reduce maintenance costs. Danish DSOs face the same challenges, as political decisions will lead to a rapid increase of EVs by 2030 (Skatteministeriet, 2020). Both short- and long-term forecast models are required to increase flexibility possibilities and understand the grid's state. In recent years, researchers have experimented with deep neural networks for accurate load forecasting. The introduction of smart meters, smart grids, and IoT devices has made it possible to gather even more data than before that researchers can use to build robust models (Amin, Cherkasova, Aitken, & Kache, 2019). Many configurations of artificial neural networks have been applied to the load forecasting problem with excellent results. The advantages of deep neural networks for load forecasting have become more evident as the data availability increases. Their inherent ability to automatically learn patterns and extract features in the data with multiple input variables makes them a suitable method to solve this problem (Koprinska, Wu, & Wang, 2018). However, because of the many different configurations and methodologies for load forecasting, they are not directly transferable to any task or application. Finally, electricity consumption data exhibits complex patterns and nonlinearity, which deep neural networks are known to be able to handle. Therefore, this thesis will investigate two key research questions:

1) How can neural networks be used to improve the forecasting of the electricity load in Danish residential areas?
2) What are the advantages or disadvantages of using deep neural networks for electricity forecasting in residential areas?

The remaining sections of this thesis are organized as follows. The literature review will determine the state-of-the-art methods for electricity load forecasting using deep neural networks. The methodology section will outline the conceptual framework and the experimental setup to find the optimal neural network for this task. The results section will present the performance of the optimal neural network against baseline and benchmark models. Finally, the results are discussed and concluded. The limitations of this research will be presented in the discussion section, while future research and recommendations are given in the conclusion.

Delimitation

This thesis is limited to electricity forecasting for Danish residential areas using deep neural networks. Thus, this study will not cover the performance of conventional statistical models for this application or literature concerning these methods.

2. LITERATURE REVIEW

The literature review for this thesis has been reviewed and accepted for publication in the *OpenSpringer – Energy Informatics Journal* as a review paper and was presented at the *1st Energy Informatics.Academy Conference*, Asia, on the 29th of May 2021.

The literature review investigates the state-of-the-art techniques using deep neural networks for load forecasting. The review presents recommendations for accurate and efficient models based on the literature analysis. The sections of this review are organized as follows. The methodology section describes the research process for finding the relevant literature. The results section presents state-of-the-art deep learning models for load forecasting and literature analysis. Finally, the results of the review are discussed.

2.1. Methodology

The literature search is performed using three online databases relevant to the energy domain. The following databases were selected: IEEE Xplore, ACM digital library, and

Elsevier (Scopus). The scoping review covers journal and research articles, conference papers, and books without any limitations to the year of publication.

Search strings were combined from prioritized keywords using Boolean operators. The combined strings were used to collect the literature from each selected database based on the contents of the title and abstract. The keyword search was restricted to the title or abstract if a large number of results were given. Table 1 summarizes the search strings and the number of results for each database conducted on the 23rd of February 2021.

Table 1 Results of keyword search in databases.

Search Strings	**Results in Database**			
	IEEE Xplore	ACM	Elsevier / Scopus	**Total**
Load Forecast AND Residential AND Neural Network	140	25	104	269
(Electricity OR Electrical) AND Forecast AND Consumption AND Neural Network	503	15	396	914
Neural Network AND (Electricity OR Electrical) AND Consumption AND Load Forecast	398	52	241	691
(Electricity OR Electrical) AND Convolutional Neural Network AND Time-Series AND Forecast	59	284	15	358
(Electricity OR Electrical) AND Long Short-Term Memory AND Forecast	393	40	158	591
(Electricity OR Electrical) AND Multivariate Time-Series AND Forecast AND Neural Networks	35	19	11	65
(Electricity OR Electrical) AND Feature Engineering AND Neural Networks AND Load Forecast	260	27	11	298
Peak Load AND Energy AND Consumption AND Forecast	193	26	155	374
				3560

The 3560 publications were imported using reference management software before removing duplicates, resulting in 2030 unique publications. The remaining references went through three rounds of filtering based on 1) keywords, 2) non-related abstracts, and 3) non-deep learning to remove non-related publications. Table 2 shows the breakdown of the filtering.

Table 2 Procedure for filtering out non-related references.

Filtering Method	Number of References	
	Non-Related	Related
1) Keyword	1597	433
2) Abstract Content	288	145
3) Deep Learning	46	99

For each of the remaining 99 publications, a full-text search was performed. For 18 references, a full-text could not be found, resulting in 81 review-related publications with full-texts. Focus aspects based on keywords were created to organize the references, which further narrowed the number of publications to an acceptable amount, which this review analyses. Table 3 shows the number of publications in each focus aspect in the reference management software.

Table 3 Number of references for each focus aspect.

Importance of accuracy	Building Consumption	Household Consumption	Demand-Side Perspective	Grid Perspective
48	23	10	19	11

2.2. Results

The amount of research conducted for predicting energy load using deep neural networks has increased substantially in recent years. With an increasing number of electric cars and appliances on the grid and the growing reliance on renewable energy sources in recent years, accurate load predictions are crucial. Researchers have experimented with various deep learning models and unique combinations to increase their performance. This section provides an overview of the applications and methods used for energy load forecasting based on the literature search.

The literature introduces two main areas in the energy domain to apply load forecasting using deep learning. Demand-side management can be defined as any utility activities that affect the demand side's load profile (Clark & Kelly, 2016). Demand-side management introduces several factors such as HVAC systems, IoT devices, lighting, or occupancy. Researchers include these aspects in their prediction models to make more accurate forecasts of a building's energy demand (Bedi, Venayagamoorthy, & Singh, 2020; Ferlito et al., 2015; Timur, Zor, Çelik, Teke, & İbrikçi, 2020).

The second main area introduced in the literature is grid control. This area focuses on understanding what happens in the grid on a short to long-term horizon from the supply side. The importance of accurate forecasts becomes especially important here as utility companies must react accordingly within a short time. In addition to energy consumption data, researchers add various weather-related data in their models to increase the accuracy (Eseye, Lehtonen, Tukia, Uimonen, & Millar, 2019; Selvi & Mishra, 2018, 2020; Torabi & Hashemi, 2012).

Several applications in the system for each main area are introduced in the energy domain. Researchers base their models on data collected from single or multiple households, public or office buildings, larger districts such as countries or states, or grids. Applications on households are divided between demand-side management and grid control depending on the use case. In demand-side management, the household's role is essential to ensure an efficient smart grid. The system can leverage demand-side management tactics based on household energy demand to shift loads, shave peaks or fill valleys (Rodrigues, Cardeira, Calado, & Melício, 2017; Saatwong & Suwankawin, 2016). Further, the household energy demand is much more volatile than an aggregated load of multiple households, meaning researchers need to consider other external inputs such as occupancy behavior, building characteristics, and even income and employment status (Ramokone, Popoola, & Awelewa, 2020; Yuce, Mourshed, & Rezgui, 2017). Regarding grid control, researchers leverage clustering algorithms on households before training their models to ensure that volatile patterns do not become an issue and increase accuracy (Aurangzeb, Alhussein, Javed, & Haider, 2021; Jarábek, Laurinec, & Lucká, 2018; Khan & Jayaweera, 2018).

Predicting a building's energy load relates to the demand-side management aspect of the literature. Researchers use predictions to open more possibilities for demand response tactics in public or office buildings. These buildings have different external inputs than households, for instance, HVAC systems, automated lighting, and occupancy patterns (Barzola-Monteses, Espinoza-Andaluz, Mite-León, & Flores-Morán, 2020; Gao et al., 2018; Katsatos & Moustris, 2019). In some buildings, a Supervisory Control and Data Acquisition (SCADA) system is installed to manage the energy consumption, supporting demand response (Vinagre, Gomes, & Vale, 2015; Vinagre et al., 2016).

In the literature, grids, smart grids, and microgrids relate to grid control and demand-side management. DSOs can use grid load predictions for more efficient energy scheduling and locating potential volatile areas in their electricity net, thereby improving their operation quality (Chan, Oktavianti, & Puspita, 2019; Shikulskaya, Urumbaeva, & Shalaev, 2020). For demand-side management, smart grids can leverage demand response measures but require accurate predictions to do so efficiently (Bruno, Dellino, Scala, & Meloni, 2018; Kaur, Kumar, Kumar, & Guizani, 2019; Krishnan, Jung, & Yun, 2020).

The following outlines the state-of-the-art deep learning and machine learning methods used in recent years. Recently, the most successful methods using neural networks have been hybrid models combining two or more neural networks, sometimes including a conventional method such as ARIMA (T. Kim & S. Cho, 2019; T. Y. Kim & S. B. Cho, 2019; Krishnan et al., 2020; Pramono, Rohmatillah, Maulana, Hasanah, & Hario, 2019; Qi, Zheng, & Chen, 2020; Rosato, Araneo, Andreotti, & Panella, 2019; Rosato et al., 2020). These models achieve very high accuracy, primarily because of the specific combination of Convolutional Neural Networks (CNNs) and Long Short-Term Memory networks (LSTM). The CNN's ability to automatically learn features (Koprinska et al., 2018) and the LSTM, with its strength in sequential data, are the basis for accurate and robust models (T. Y. Kim & S. B. Cho, 2019; Qi et al., 2020; Rosato et al., 2019). The hybrid models all require multivariate inputs to exploit the combined neural networks' capabilities. Weather and calendar days data inputs are standard among these methods.

In recent years, researchers have used LSTM models widely with excellent results. However, the models lack the ability to learn features, requiring extensive feature engineering beforehand (Al Khafaf, Jalili, & Sokolowski, 2019). Also, CNN's have been performing well in their predictions, though they have some difficulties in volatile consumption patterns (Aurangzeb & Alhussein, 2020). Thus, the two neural networks complement each other very much in a hybrid model. Neural networks have also been applied to other domains, such as weather forecasting with success (Wollsen & Jørgensen, 2015), proving that neural networks can be used for other areas that use time-series-based data.

Researchers have carried out unique approaches such as ensembling conventional models with neural networks, boosting, decomposition measures, and clustering algorithms with good performances (Ai, Chakravorty, & Rong, 2020; Bot, Ruano, & da Graça Ruano, 2020;

Chenglei, Kangji, Guohai, & Lei, 2015; Rahman, Alam, & Rahman, 2019; Ves et al., 2019). Additionally, researchers often compare the deep learning models to conventional methods such as autoregressive integrated moving average (ARIMA). While the conventional methods perform well, the neural networks outperform them when it comes to load forecasting (Kuo & Huang, 2018; Panapongpakorn & Banjerdpongchai, 2019). Therefore, some neural networks are combined with conventional methods to improve performance. Building hybrid models increases performance metrics substantially compared to the non-hybrid subparts (Chan et al., 2019; Krishnan et al., 2020).

Table 4 presents a collection of the models that researchers have experimented with. Further, the data source is included to provide some context of the applications and give a foundation for comparing the experimental results. The evaluation criteria selected are RMSE, RMSPE, MAPE, and MAE. The forecast horizon is also added to outline the relationship between the performance and the future prediction point. The table presents varying results for many different configurations of neural networks presented earlier in this section. In particular, it shows how well the hybrid CNN-LSTM performs when just looking at evaluation metrics. However, evaluation metrics should not be the only deciding factor for choosing the model. The data context and forecast horizon significantly impact the networks' performance. This table only includes publications that were analyzed earlier in this section. Publications that did not use the evaluation metrics displayed in the table were not included.

Table 4 Selected models including their performance and data source.

Reference	Model	RMSE / RMSPE	MAPE / MAE	Horizon	Data Context
(Katsatos & Moustris, 2019)	MLP	20.5 kWh	16.4%	24hrs	Data collected from public building in Athens, Greece
(Koprinska et al., 2018)	CNN	476.9 kW / 2392.88 kW / 642.52 kW	340.64 kW / 1884.86 kW / 497.62 kW	24hrs	Publicly available data from Australia, Portugal, and Spain
(Ferlito et al., 2015)	NARNN	15.7% – 17.97%	-	3mo – 12mo	Data collected from public building in Eboli, Italy
(Timur et al., 2020)	RBF	0.33 MWh	11.83%	1mo	Data collected from hospital building in Adana, Turkey
(Selvi & Mishra, 2018)	Unspecified ANN	-	2.9%	1hr	Data collected from DSO in Delhi, India.

(Torabi & Hashemi, 2012)	Unspecified ANN	-	1.96%	1hr	Data from Bandar Abbas, Iran
(Eseye et al., 2019)	Unspecified ANN	-	1.96%	24hrs	Data collected from buildings in Espoo, Finland.
(Aurangzeb et al., 2021)	CNN	-	39%	24hrs	Data collected from households in Australia for Smart Grid Project
(Jarábek et al., 2018)	LSTM	-	15%	24hrs	Data collected from enterprises in Slovakia.
(Khan & Jayaweera, 2018)	Unspecified NN	-	5.85% – 16.25% / 7.92 kWh – 22.93 kWh	24hrs	Data collected from smart meters in Ireland.
(Barzola-Monteses et al., 2020)	LSTM	5.085 kW	3.714 kW	24hrs	Data collected from public building in Guayaquil, Ecuador
(Vinagre et al., 2015)	MLP	-	13.6%	5min	Data collected from office SCADA system
(Rosato et al., 2019)	LSTM	2.252 kW – 7.061 kW	-	24hrs	Data collected from power plant in Denver, CO
(Qi et al., 2020)	CNN-LSTM	-	1%	24hrs	Data collected from industrial area in China.
(Pramono et al., 2019)	CNN-LSTM	203.23 kW	2.02% / 142.23 kW	1hr	Data collected from public datasets (New England, USA & Switzerland)
(T. Y. Kim & S. B. Cho, 2019)	CNN-LSTM	0.6114 kW	34.84% / 0.3493 kW	1hr	Data collected from UCI ML repository on households
(Al Khafaf et al., 2019)	LSTM	-	3.15%	3day	Data collected from households in Victoria, Australia
(Ves et al., 2019)	Ensemble	6.19 kWh	1.59% / 5.60 kWh	4-24hrs	Publicly available data UK-DALE
(Bot et al., 2020)	Ensemble	0.8 kW – 1.6 kW	-	12hrs	Data collected from Smart Home in Davis, USA
(Ai et al., 2020)	Ensemble	-	0.0199%	24hrs	Data collected from households in Norway.
(Chenglei et al., 2015)	PSO-ANN	-	0.0169% - 0.0606%	24hrs	Data from buildings in America

The models used in the literature are not unique to any specific energy domain aspect. The main focus is to increase the models' accuracy, where researchers leverage many different methods as described in the previous section. While non-hybrid models perform very well, the literature suggests that hybrid models are more often applied and more successful. The main argument for hybrid models is how they complement each other, where the other lacks. LSTM networks and CNNs are the most common neural networks for load forecasting using deep learning, both as hybrids and non-hybrids.

The forecasting horizon is an essential factor to remember when comparing the models' performances. From Table 4, we can see that the models' errors range widely depending on the forecasting horizon and the data source. Based on this, before choosing the desired model, the use cases should be considered beforehand. It is not enough to compare the forecasting horizons because the datasets can impact the model's ability to predict validation data. For instance, the models from (Pramono et al., 2019) and (T. Y. Kim & S. B. Cho, 2019), which use the hybrid architecture and forecasting horizon of one hour, show substantially different results. While there can be many factors impacting this performance, the data inputs undoubtedly influence the error rate.

The models are applied from households to buildings and grids to whole countries. The literature suggests that buildings are heavily related to demand-side management, whereas grid and household applications can be seen from a grid and demand-side perspective. The significant challenge for households is the volatility in the load profile. Researchers circumvent this by clustering households into similar profiles and aggregating them.

For building applications, exogenous data such as HVAC systems or lighting is essential. These factors can leverage demand response for load shifting and increase the accuracy of the load demand for the building. Grid applications can be used to schedule the energy supply better, increasing the operation quality of DSOs. From a demand-side perspective, accurate forecasts make demand-response in smart grids more relevant as there is more certainty of the grid's future state. Each application requires multivariate inputs for an accurate forecast based on the literature.

Researchers who use more relevant variables received well-performing models. However, the literature did not cover the limitations of applying a high-parameter model to a real-world scenario. Increasing relevant input variables may increase the accuracy, but the models may not be efficient enough that DSOs can rely on their prediction on a short-term basis. Furthermore, increasing the model's complexity by ensembling multiple models or other

unique approaches is not necessarily a good idea in a real-world scenario. It is especially true if a more straightforward model performs at the same level as the complex model.
The main areas introduced in the first section share the primary goal of making as accurate forecasts as possible. However, they differ in their real-world applications. Demand-side management uses accurate forecasts to leverage demand response measures, or in other ways managing the demand in households, buildings, or smart grids. On the other hand, grid control uses forecasts to understand what will happen in the energy system on short to medium-term horizons and react accordingly from the supply side. Two considerable challenges for both areas are volatile load profiles and uncertainties in consumption. The literature suggests increasing feature engineering and data inputs to understand the variations, seasonality, and load profile patterns.
The previous sections outlined the main areas for load forecasting using deep learning techniques and the state-of-the-art methods applied to households, buildings, and grids. Hybrid-based models for all aspects of the energy domain were most successful, outperforming many conventional and single-type neural network models. The literature suggests the dominating hybrid model be a CNN-LSTM because of its ability to learn features and sequential data, respectively. Researchers have used non-hybrid CNNs and LSTM networks to successfully forecast electricity consumption, emphasizing the advantages of these two neural networks. While conventional methods perform well, most neural network models outperform these by a large margin, underlining neural networks' potential for load forecasting. The use of multivariate data is an essential factor in achieving accurate models. Depending on the application, such data could be the weather, calendar days, indoor climate, HVAC systems, and occupancy behavior. Thus, based on the research carried out for the literature review, it is recommended to use a convolutional-recurrent neural network hybrid (CNN-RNN) for the most accurate and efficient load forecast. The hybrid model has advantages over non-hybrid models, such as increased sequence and pattern recognition in the data. However, it is impossible to decide on a single architecture for every use case because the performances vary depending on the forecasting horizon, data sources, and application scenarios. These factors must be analyzed and tested beforehand to find the most suitable model for the scenario. The literature suggests that a hybrid neural network should be trained on day and time features, weather data, and historical consumption data as a minimum. Finally, if working with household consumption,

it is suggested to aggregate the load of all households or create aggregated clusters of similar households.

2.3. Discussion

The review presents many experiments using deep neural networks for load forecasting in recent years because of the increasing amount of data available. The results indicate that hybrid models consisting of convolutional and recurrent neural networks are the state-of-the-art methodology for accurate load forecasting. Further, multivariate inputs are a standard among deep learning applications. Compared to conventional methods, deep neural networks are in many ways more advantageous for load forecasting.

In the literature, hybrid models are compared to conventional methods and single neural networks with remarkable results (Chan et al., 2019; Dudek, Pełka, & Smyl, 2020; Krishnan et al., 2020). Models must be accurate throughout the year because DSOs depend on the predictions for their grid operation. Thus, improvements to neural networks' configurations must be made continually to make these models applicable in real-world scenarios. Moreover, the literature divided applications between demand and supply, suggesting different ways of applying deep neural networks for load forecasting within each perspective. Demand-side purposes often require much different input data than the supply-side. Demand-side management aims to somehow affect the load without supplying more or less energy. Therefore, systems must be in place for that control. DSOs can react to the load by down- or up-regulating energy on the supply side. Meaning, a low error margin is required as many inaccurate predictions could result in power outages. (Clark & Kelly, 2016)

The scoping review results fit very well with the rising trend of using deep neural networks to solve problems. While previous research has focused on improving conventional methods such as ARIMA for load forecasting, this review demonstrates that the applications of recurrent and convolutional networks for time series data are outperforming them. This literature review confirms the strengths of the recurrent neural networks for time series data as a high number of publications used these for their ability to learn sequential data (Hrnjica & Mehr, 2020). The results also build on the existing evidence that convolutional neural networks are used extensively because of their ability to learn repeating patterns and features in the data automatically (Koprinska et al., 2018). Based on the analyzed literature, a mapping of the focus aspects and their applications can be made, as shown in Figure 1. The figure presents the separation of the demand-side applications from the supply-side. As

outlined in the scoping review results, public and office buildings primarily relate to demand-side management when it comes to deep learning load forecasting. A slight overlap to the supply side is visualized as this scoping review does not cover all publications.
Further, smart grid applications primarily relate to demand-side applications because they can leverage demand-response measures in these grids. On the supply side, households are mainly covered. However, the exceptions are households that utilize energy management systems related to demand response. The predictions made for grid applications are visualized on the supply side. Here, grid operators can react to any electric load change by regulating energy. Finally, the most common input features for each application are displayed. There are significant differences in the deep neural networks' input data, depending on the focus aspect. Future research must emphasize the data's inputs and context for their results to be comparable.

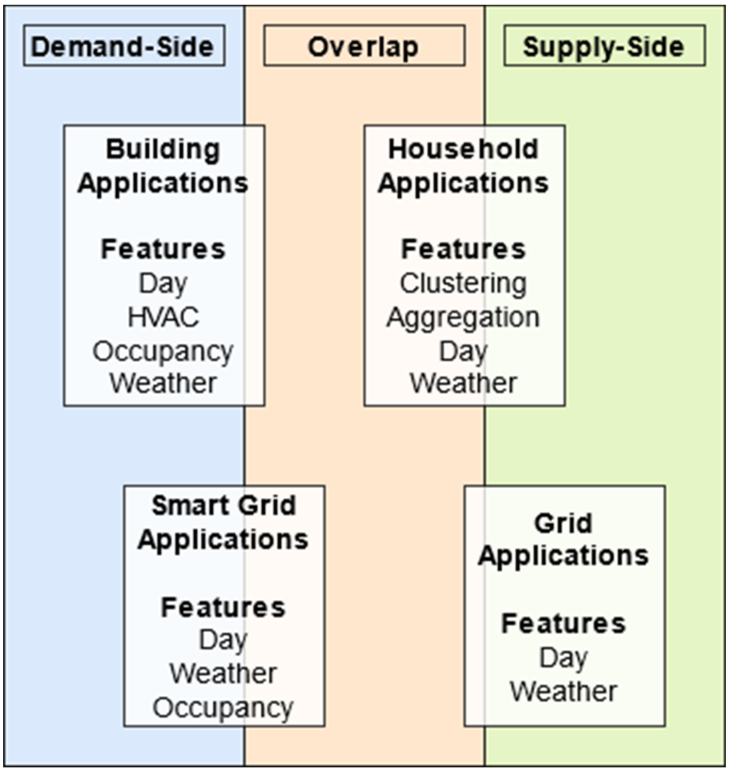


Figure 1 Mapping of the focus aspects and the applications

This review's limitations are the context and data sources in which the many models are applied. The amount and type of data used to train the models may impact their performance, limiting the generalizability. Various geographical regions may show different consumption patterns depending on the societal background. For instance, the performance in (Barzola-Monteses et al., 2020) on a public building is difficult to compare to the performance of many households' load as in (T. Y. Kim & S. B. Cho, 2019). However, the significant findings from such a comparison are the model's architecture and configuration. Furthermore, this review suggests a difference in model performance if the consumption data is taken from a household, public building, office building, or grid. Therefore, a thorough analysis of each data source and its context is required, and controlled testing on the forecasting horizons with different datasets is needed. However, this is beyond the scope of this review.

Finally, this literature review narrowed the search to deep learning techniques, leaving out publications focusing heavily on conventional methods. Instead, conventional machine learning methods and deep neural networks are compared in publications that benchmark the models against each other, as seen in (Panapongpakorn & Banjerdpongchai, 2019). Consequently, this review does not consider research focusing solely on conventional methods.

3. METHODOLOGY

The methodology chapter outlines the experiments conducted for this thesis. Firstly, the theory behind neural networks and why they are useful for this study will be explained. Then, the evaluation metrics will be described, followed by the experimental setup and feature selection & engineering decision-making. Finally, the hyperparameter tuning and final proposed model will be presented.

3.1. The Neural Networks

This section will briefly explain the concepts of Convolutional and Recurrent Neural Networks. Based on the literature review, two potential CNN-RNN architectures are chosen for the final hybrid model. The first combines a convolutional layer with a Long Short-Term Memory network. The other combines a convolutional layer with a Gated Recurrent Unit. The final decision is based on the performance in the hyperparameter and network configuration experiments.

3.1.1 Convolutional Network

Convolutional Neural Networks were first seen applied to recognizing handwritten digits (LeCun et al., 1989) and documents (LeCun, Bottou, Bengio, & Haffner, 1998). The network architecture is based on the visual cortex of the biological eye, which has neurons with local receptive fields that react to smaller parts of the visual field (Hubel & Wiesel, 1968). For

CNNs, the mathematical translation of the visual cortex corresponds to the convolution operation (Goodfellow, Bengio, & Courville, 2016). This operation learns local patterns of the data instead of looking at the whole range of available data. Thus, the convolution operation acts as a dimension reduction that learns the data segments and pieces them together (Chollet, 2018). While CNNs show their strength in computer vision that uses two-dimensional image data, they are equally applied to one-dimensional data such as time-series sampled at regular time steps (Goodfellow et al., 2016). A convolutional block usually consists of three main components starting with the convolution operation, an affine transformation of the inputs. A nonlinear activation function follows, typically a rectified linear unit. Finally, pooling is applied to the output, reducing the dimension further (Goodfellow et al., 2016).

For this thesis, the convolutional layer will act as a feature extractor for the subsequent recurrent layer. Based on the literature, the CNN should not follow the recurrent layer but can be used as the only type of neural network for load forecasting if multiple blocks of convolutional layers are applied (Aurangzeb et al., 2021). The results from the literature review indicate that a CNN excels at extracting useful patterns and information from the input, which should improve the recurrent layer’s ability to learn the sequential patterns. The specific parameters and configuration for the CNN will be decided by hyperparameter tuning later in this chapter. Figure 2 illustrates the convolutional layer for the hybrid neural network. In the input column, the dashed square area signifies the input for the convolutional layer. This window has the size of the lookback times the number of features. The lookback determines how far back in time the model will look from the prediction timestep, while the number of features are the multivariate input. After the convolution layer has done its operation, the local patterns will move into the pooling layer, reducing the size by subsampling the maximum values in a one-dimensional grid with the size of two. Finally, the output can either be fed into another block of convolution and pooling operations or move into the hybrid model's recurrent block.

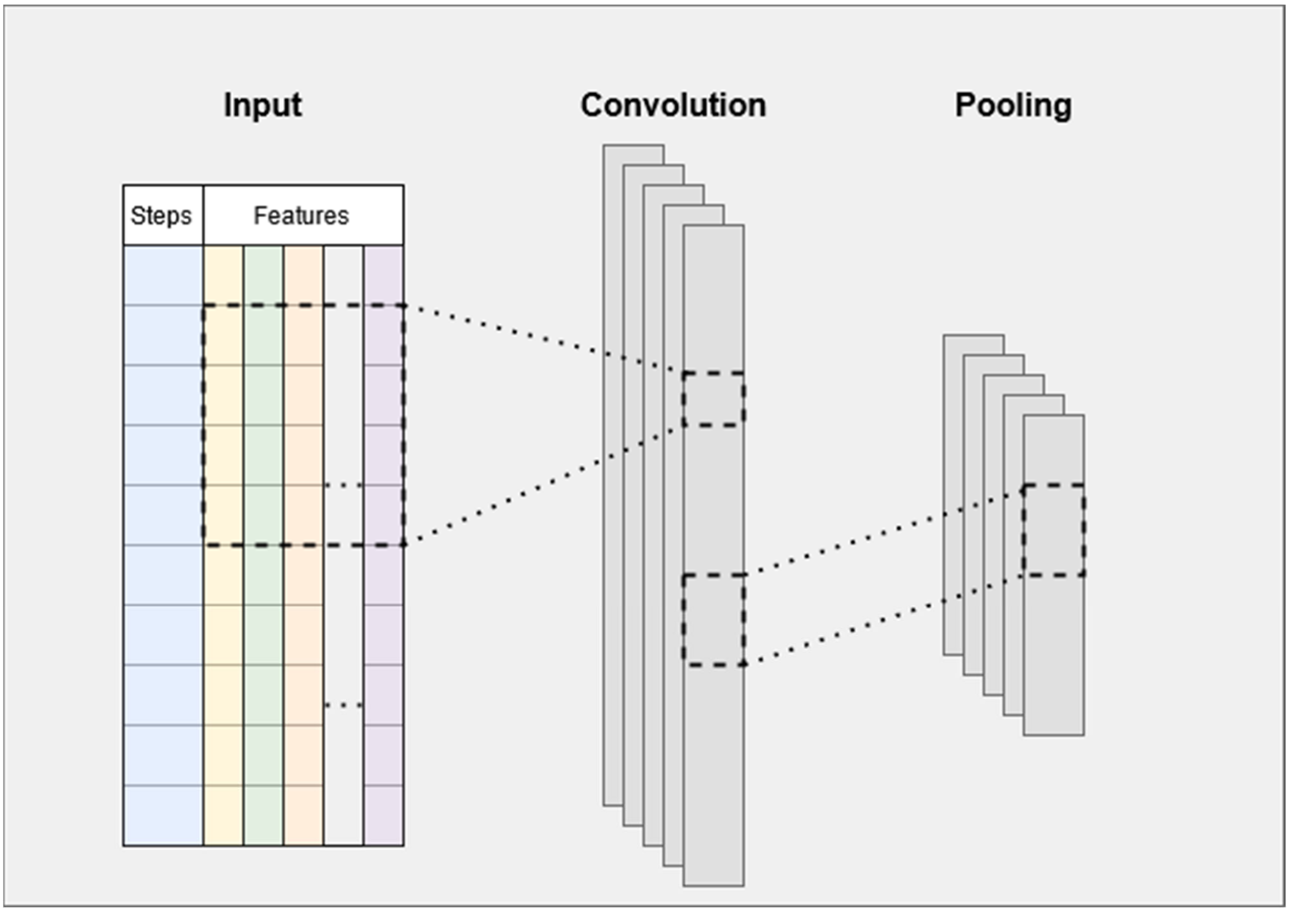


Figure 2 Flowchart of the convolutional block.

3.1.2 Recurrent Network

Recurrent Neural Networks specialize in sequential data by backpropagating through time (Rumelhart, Hinton, & Williams, 1986). However, this strategy will suffer from short-term memory problems when the gradients either explode or vanish during training (Hochreiter & Schmidhuber, 1997). For this reason, two versions of recurrent networks emerged as the primary applications for longer sequences of data. The Long Short-Term Memory network was proposed in 1997, and the Gated Recurrent Unit was proposed in 2014. Both deal with the unstable gradients problem by introducing custom cells to the network. An LSTM cell (Hochreiter & Schmidhuber, 1997) takes advantage of hidden states and consists of forget, input, and output gates that decide what information to transfer. The forget gate erases information from memory, the input gate decides what to add to the long-term memory, and the output gate decides what should be transferred to the short-term state and to the output at the time-step. Each gate's decision is made by the sigmoid activation function, which transforms the values to be between 0 and 1. Furthermore, the hyperbolic tangent activation functions helps regulate network by transforming the values to be between -1 and 1. The GRU cell (Cho et al., 2014) is similar to the LSTM cell, but merges the operations of the input and forget gate into one. The merged input-forget gate updates or resets the information to be passed on using a sigmoid activation function. The output gate has no role in the GRU cell, and the entire state is sent to the next cell through a hyperbolic tangent activation function. Figure 3 illustrates a simple internal operation of an LSTM and GRU cell.

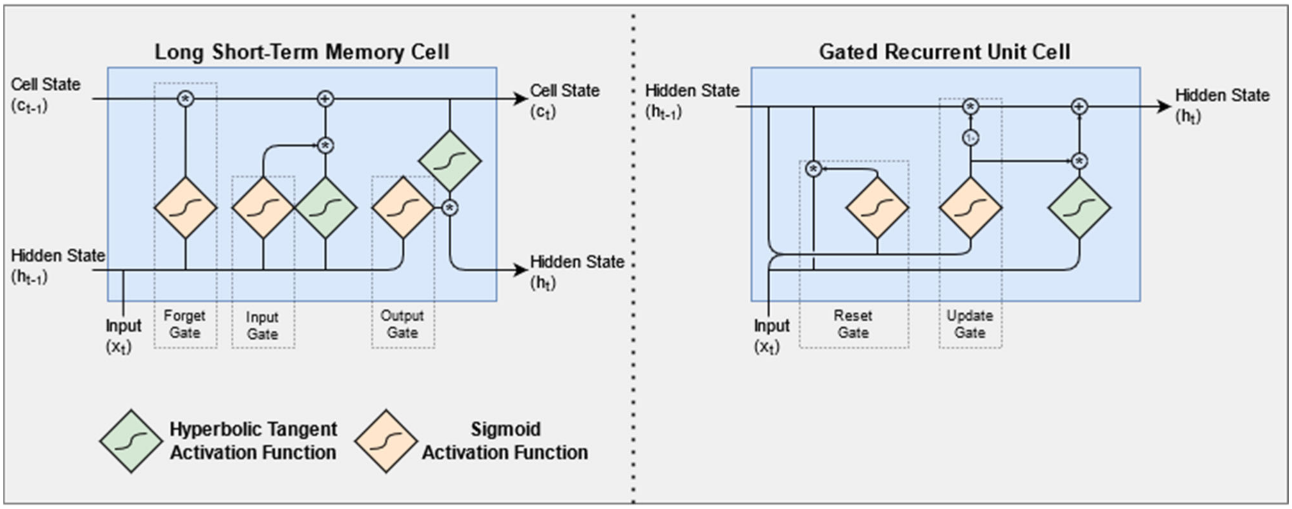


Figure 3 Flowchart of the LSTM and GRU cells. Inspired by (Géron, 2019).

RNNs are particularly advantageous for the experiments in this thesis, as the electricity load is time-series data with repeating complex patterns. The hidden states and long-term memory will help learn longer sequences of consumption data. Because of the similarities of the two cells, it will not be decided beforehand which type of RNN will be chosen for the hybrid model. Instead, each type will be tested during the hyperparameter tuning and architecture configurations. However, as the literature suggests, it is evident that LSTM networks are used more often for electricity load forecasting applications. Nevertheless, this thesis will test each RNNs performance before deciding on the final hybrid model.

3.2. Evaluation Metrics

To evaluate the performance of the hybrid model, the baseline and benchmark results of non-hybrid neural networks, three error metrics will be used. Y are the true values, and $\hat{Y}$ are the predicted values.

3.2.1 (Root) Mean Squared Error

The Mean Squared Error is used as the loss function during model training, and its root will be used as one of three evaluation metrics on the final predictions. Equation 1 shows the mean squared error. Equation 2 shows the root mean squared error.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \qquad \text{Equation 1}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2} \qquad \text{Equation 2}$$

3.2.2 Mean Absolute Error

The mean absolute error is included as an evaluation metric as it can be easier to interpret the error value from this metric than the MSE. The MAE will be tracked during model training with the MSE and calculated for the final predictions. Equation 3 shows how the mean absolute error is calculated.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right| \qquad \text{Equation 3}$$

3.2.3 Mean Absolute Percentage Error

The mean absolute percentage error is often used for forecasting, and it describes how accurate a forecast model is as a percentage. This metric is not used during training because the variables are all standardized, meaning some zero-division may occur. Thus, after inverse-scaling the predictions and true values, the MAPE can be calculated as in Equation 4.

$$MAPE = \frac{1}{n}\sum_{i=1}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| \qquad \text{Equation 4}$$

3.3. Experimental Setup

The experimental setup section outlines the systematic process and decision-making to find the optimal proposed model and how the data inputs were prepared and analyzed. Firstly, the pipeline from data collection to processed data will be described. Afterward, the analysis of the data and the feature selection & engineering steps are illustrated. Finally, the proposed model is presented with the results of the hyperparameter tuning.

3.3.1 Data Source

The datasets used in this thesis consist of real-world energy meter readings gathered in conjunction with the nationwide project Flexible Energy Denmark (FED). FED is a data-driven research project that aims to develop digital solutions to accelerate the green transition by increasing flexibility in the energy sector (Flexible-Energy-Denmark, 2019). The danish distribution system operator TREFOR has two test areas of their grid designated as living laboratories for the project, located in Strib, Middelfart, and Nørre Bjert, Kolding. Electricity consumption readings are collected on an hourly basis for most installations within the two areas. However, some meters collect data at a quarter-hourly interval. Each test area is connected to one substation, which will be the level of aggregation used in this thesis. The datasets contain 164 and 136 installations for the Kolding and Middelfart test areas, respectively, with data ranging from January 1st, 2019, until May 4th, 2021. Figure 4 shows the data pipeline from the source to the processed dataset ready for model training. Additionally, the grids for each test area can be seen in **Error! Reference source not found.** and **Error! Reference source not found.**.

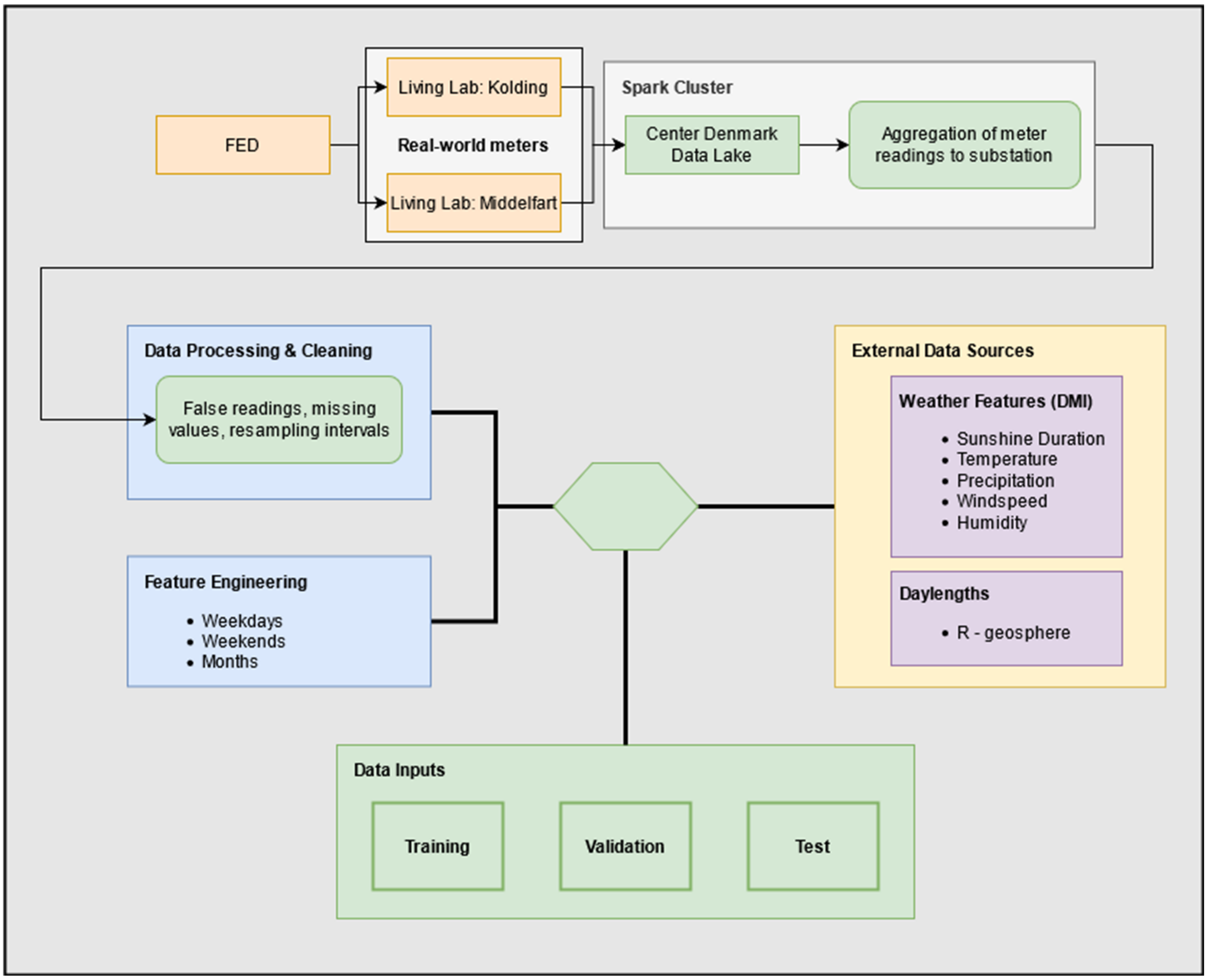


Figure 4 Data pipeline from raw to model inputs.

3.3.2 Data Processing & Analysis

Because the meter readings are non-validated real-world data, some cleaning and processing are required before model training. Some non-validated meter readings are prone to having "sick meters" or meter resets, meaning they will sometimes have large positive or negative values. The data had around five days of missing values in each dataset. These missing values were filled using data from the same day of the previous year. Moreover, the DSO sends adjustments of meter readings with up to a 3-month delay, requiring adding or subtracting the adjustments on duplicate timestamps. The data were then resampled to 1-hour intervals to account for the quarter-hourly meter readings, which resulted in the response variable of the combined load for each area's substation at an hourly interval. The data processing resulted in two datasets with 20520 timesteps in each.

3.3.3 External Data Sources

Multiple variables are used from external sources to improve the model's predictions by enhancing the context of the dependent variable, including weather data and day lengths. Weather data collected from the Danish Meteorological Institute (DMI) will be used as independent variables for the neural network. The DMI weather station closest to both test areas in Assens, Funen, was found using longitude and latitude coordinates. The weather station is roughly 30 kilometers away from each area. Initially, six weather parameters were collected. Sunshine minutes in the last hour, dry temperature, relative humidity, wind speed, precipitation duration, and precipitation amount were collected from the DMI API from January 1st, 2019, to May 4th, 2021. The day lengths for the time range were collected using the R programming language library *geosphere*. The code for this can be found in Appendix B.

3.4. Feature Selection & Feature Engineering

This section describes the decisions made to select the most useful among the many variables and create new features based on dates and time. For each variable, some assumptions have been made supported by data evidence. The following features will be discussed in detail: historical electricity load, sunshine minutes in the last hour, dry air temperature, day lengths, weekdays, weekends, and months. Some initial testing on a neural network showed that including all six weather features added too much noise to the data and harmed its performance on validation and test data. Thus, the weather features were reduced to only include sunshine minutes in the last hour and dry air temperature. These two variables should do a decent job capturing the seasons and the general weather pattern without including the other redundant variables. Each variable will be analyzed with the electricity load value in the following sections, and the assumptions for choosing each specific feature are outlined.

3.4.1 Electricity Load

The first feature, which is also the response variable for the neural network, is the historical load of each test area's substation since the beginning of 2019.

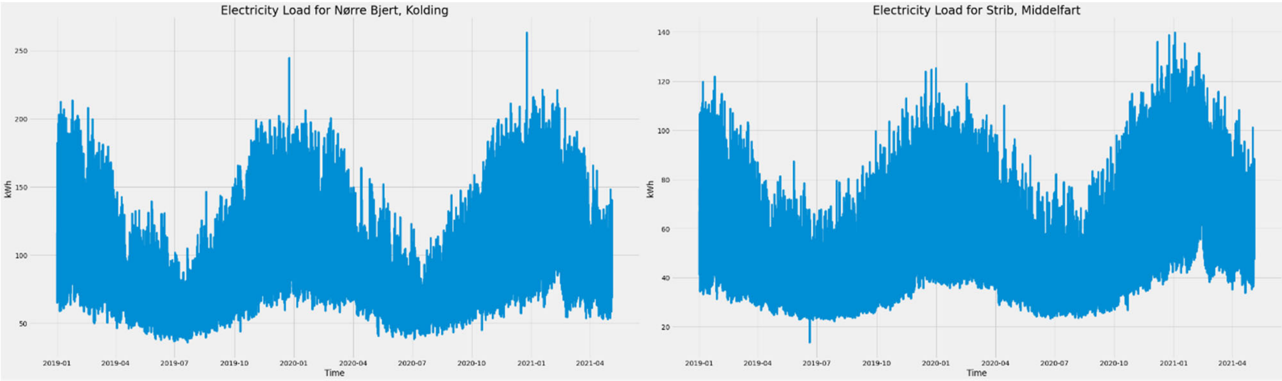


Figure 5 Historical electricity consumption for each Living Lab.

Figure 5 displays the load for each area. A particular seasonal pattern is evident in the data, with lower hourly kWh readings during the summer and higher readings during the winter. Figure 6 compares the last week in January 2019 to the last week in August 2019. While the pattern throughout the day stays the same, the amount of energy consumed is substantially higher during a winter week than during a summer week. Thus, the model will need time features to give context to the energy consumption.

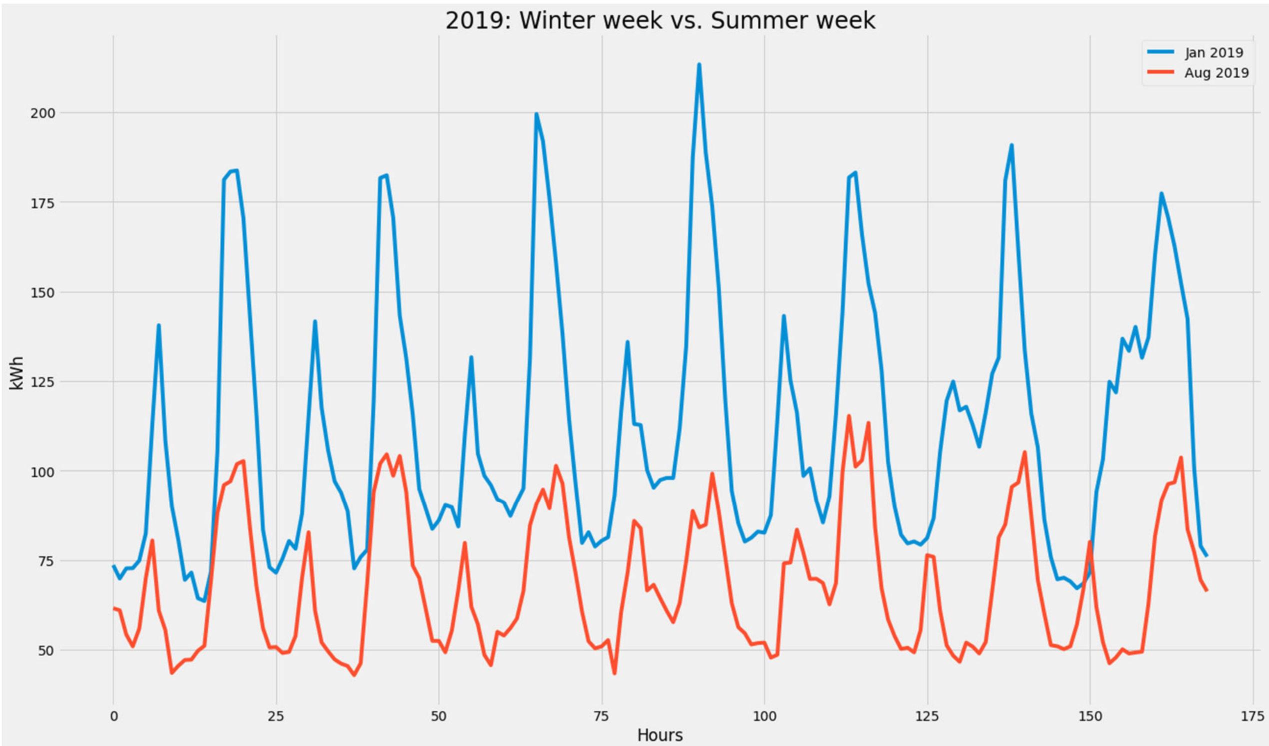


Figure 6 Seasonality differences for the electricity consumption.

Additionally, 2020 and 2021 have been affected by COVID-19 lockdowns, which may have impacted energy consumption during this time and changed some behavioral patterns.

However, it is difficult to determine anything about the impact of COVID-19 on the two test areas with the limited amount of data available.

Table 5 Descriptive statistics for the electricity load feature.

	Count	Mean	Std	Min	25%	50%	75%	Max
Kolding	20520	89.87	32.08	35.9	66.28	82.22	106.84	263.16
Middelfart	20520	54.46	19.24	13.52	40.14	51.21	65.5	139.79

Table 5 shows the descriptive statistics for the feature from each dataset. The overall higher energy consumption in Kolding compared to Middelfart is partly due to a higher number of installations in Kolding. On average, one installation in Kolding consumes ≈0.15 kWh more than one installation in Middelfart on an hourly basis. This difference in consumption could relate to a higher proportion of installations with heat pumps and electric heating in the Kolding data than in the Middelfart data. Looking at just the top ten consumers in the 2019-2020 period, nine out of ten installations in Kolding have either heat pumps or electric heating installed. At the same time, just three of the top ten consumers in Middelfart have heat pumps installed, meaning the load pattern is slightly different in each dataset, as the heat pumps and electric heating will be more visible in the Kolding dataset. When comparing the results of the neural networks, this difference in the two datasets should be considered.

3.4.2 Sunshine Duration

The general assumption for using sunshine duration as an independent variable is that residential energy consumers spend less energy when the sun is shining. The aim is to extract behavioral patterns of the residents, signaling warmer and lighter days with possibly less energy consumption and vice-versa.

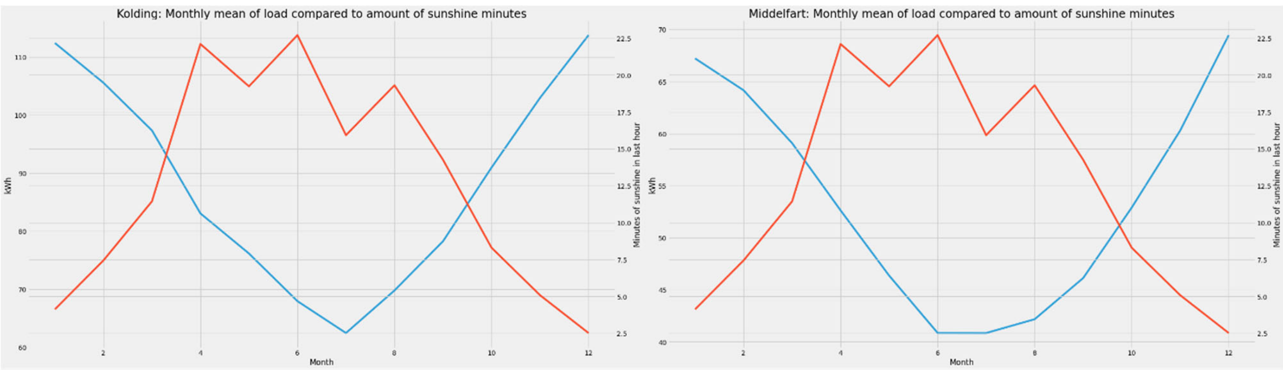


Figure 7 Monthly mean of load compared to the sunshine duration.

Figure 7 indicates negatively correlated features as the mean electricity load decreases during spring and summer months, the mean sunshine duration increases. Using the sunshine duration as a feature can give some more context and seasonality to the electricity load and hopefully extract meaningful patterns that help the prediction. Table 6 shows descriptive statistics for the sunshine duration variable. The highly skewed distribution of the variable relates to the general weather in Denmark and due to nighttime periods.

Table 6 Descriptive statistics for the sunshine duration.

	Count	Mean	Std	Min	25%	50%	75%	Max
Sunshine Duration (mins/hr)	20520	12.53	21.52	0.0	0.0	0.0	18.0	60.0

3.4.3 Dry Air Temperature

The assumption for using the temperature is based on the changing behavior of people depending on the outside temperature. For instance, a warm spring day and a cold summer day may show very distinct patterns in energy consumption. Similar to the sunshine duration, the temperature shows a negative correlation with the electricity load, as seen in Figure 8. From the figure, we can also see an outlier period of low temperatures at the beginning of 2021, which is not seen in the previous years. This outlier must be considered when using the data for predictions.

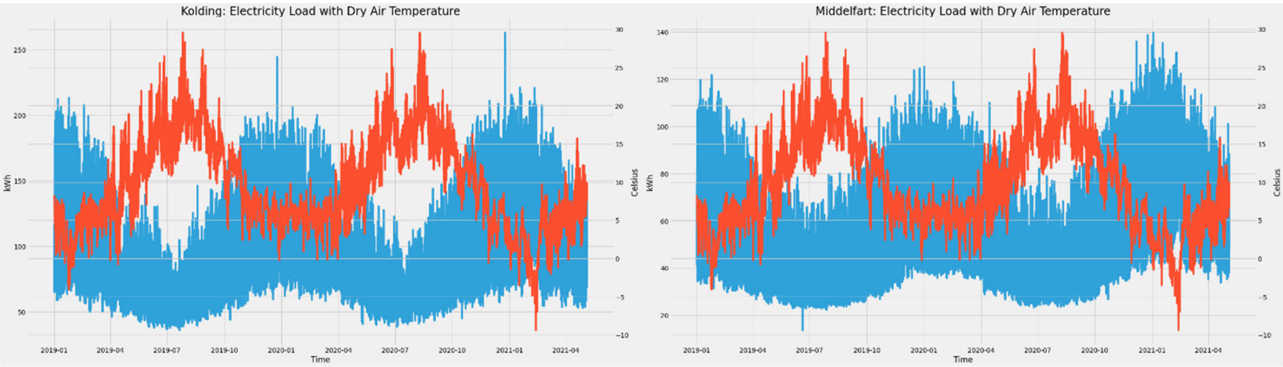


Figure 8 The electricity load and dry air temperature.

Table 7 shows descriptive statistics for the temperatures measured at the Assens, Funen weather station. It is a reasonably regular distribution of temperature measurements for Denmark and does not look out of the ordinary, except for the minimum value measured of -9.42 degrees Celsius.

Table 7 Descriptive statistics for the dry air temperature feature.

	Count	Mean	Std	Min	25%	50%	75%	Max
Temperature (C)	20520	9.46	6.04	-9.42	5.0	8.03	14.17	29.57

3.4.4 Day Lengths

The day length feature should extract recurring seasonal patterns based on the length of the day. The winter in Denmark has short days, with the shortest being around seven hours long, while summers can have much longer days lasting up to almost 18 hours. The assumption is that shorter days result in higher energy consumption as residents spend more time inside, use more electrical appliances and lights in their homes than during the summer, e.g., using tumble dryers instead of washing lines outside. Figure 9 shows the day length period with the electricity load from Kolding and indicates a negative relationship between the variables supporting the assumption.

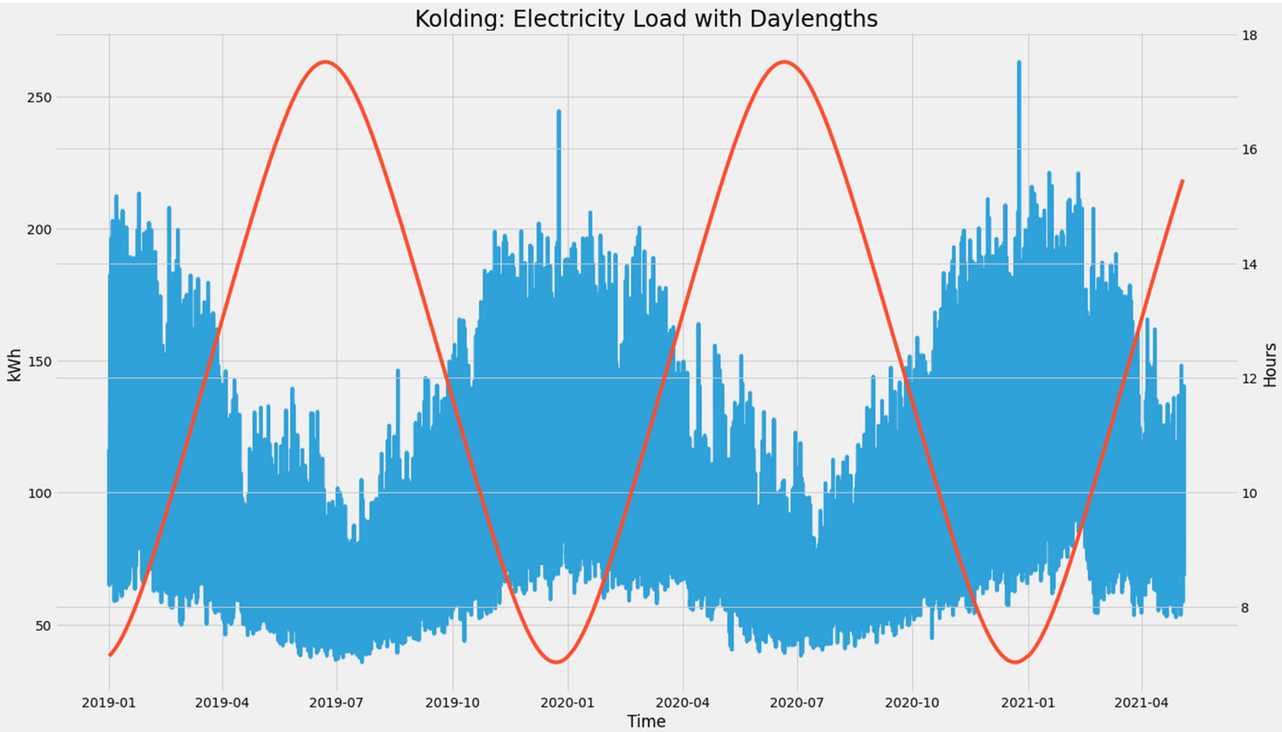


Figure 9 The electricity load and day lengths.

3.4.5 Date & Time Features

The following section outlines three different variables based on the timesteps created to contextualize the difference in days and months.

The first engineered feature is the weekday variable ranging from 0 to 6 (Monday to Sunday). The assumption is that daily consumption is not the same, and there can be varying patterns throughout a week. Figure 10 compares a week's average load during a summer period with a winter period. It is evident that weekdays differ throughout the seasons, and there seems to be a trend towards an increasing load as the weekend gets nearer.

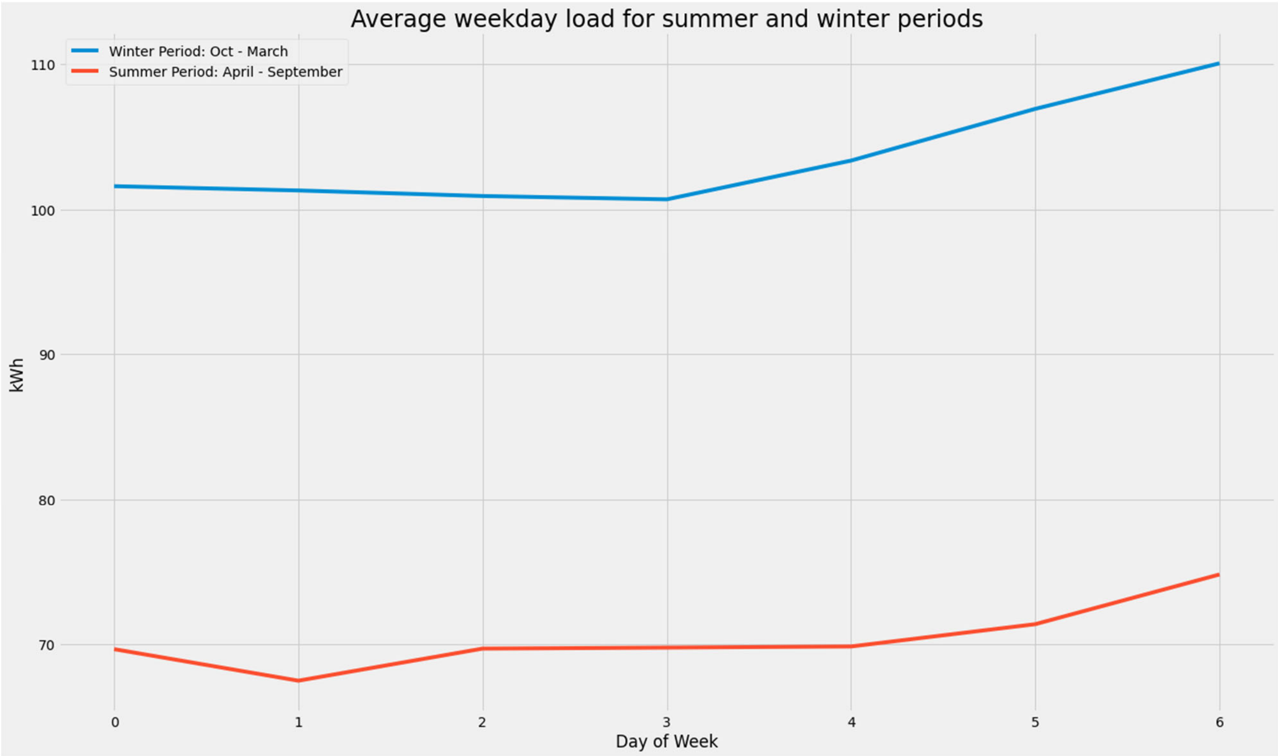


Figure 10 Mean weekday load for summer and winter periods.

However, to better encode the cyclical nature of the days and not confuse the neural network with integers that could indicate that Sunday is greater than Monday (6 > 0), the values are transformed into sine and cosine transformations that better capture the cyclic variable. The weekday variable's combined sine and cosine transformation can be seen in Figure 11. By transforming the variable into two dimensions, Sunday is now closer to Monday mathematically than when using integers. The sine and cosine transformations are done by using Equation 5 and Equation 6.

$$x_{sin} = \sin\left(\frac{2 * x * \pi}{\max(x)}\right) \quad \text{Equation 5}$$

$$x_{cos} = \cos\left(\frac{2 * x * \pi}{\max(x)}\right) \quad \text{Equation 6}$$

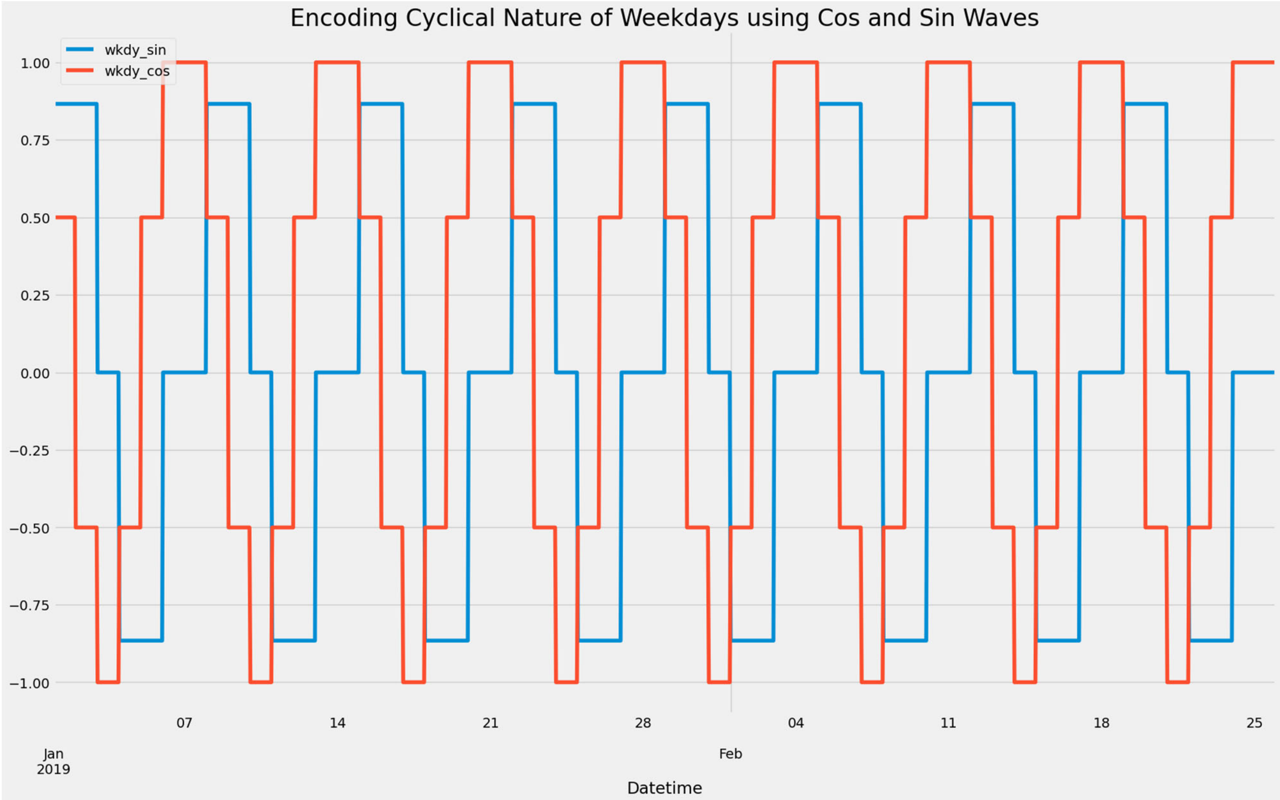


Figure 11 Sine and cosine transformations of weekday integers.

The second engineered feature relates a lot to the weekday feature—a binary variable for indicating if a meter reading is measured during the weekend or not. Figure 12 shows the increasing average electricity load on the weekends. The feature will separate business days from weekends and extract information about the patterns during these periods.

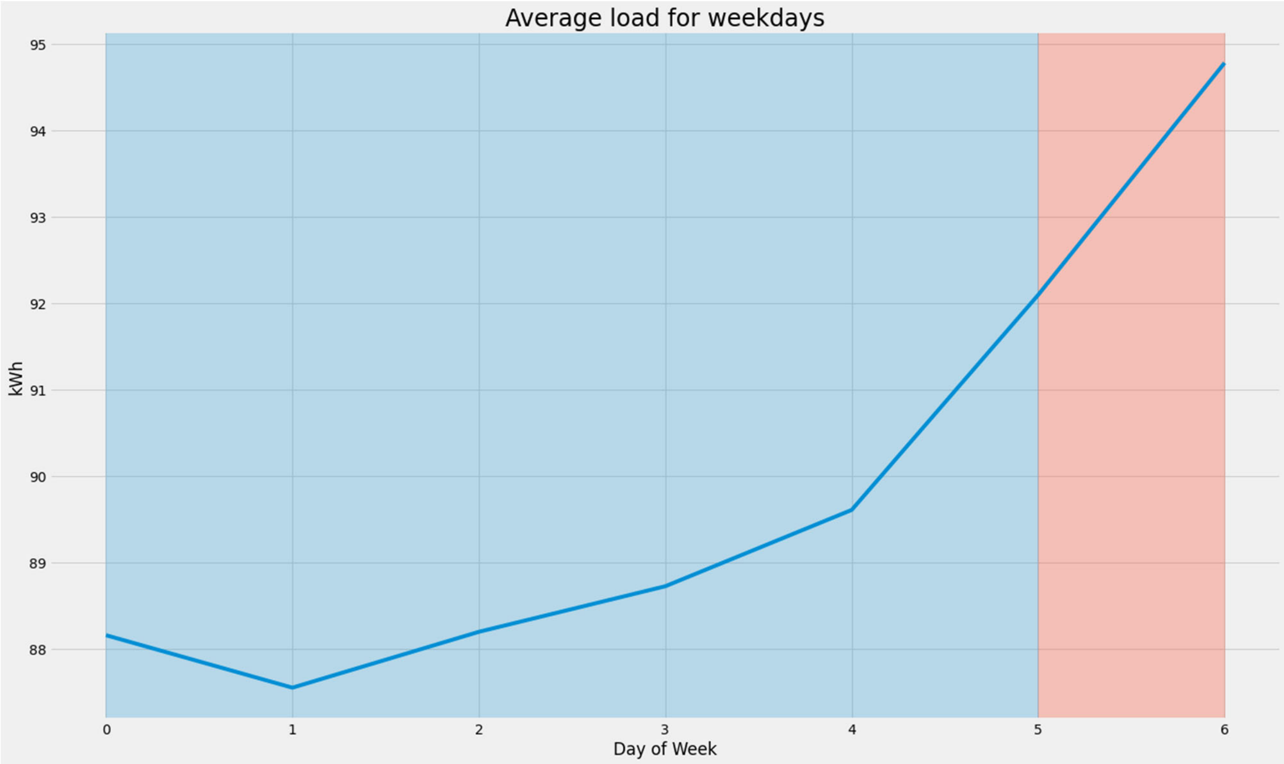


Figure 12 Mean electricity load per weekday.

The final engineered feature is the month variable, which should reinforce the seasonal pattern in the data. As seen throughout the feature selection section, the electricity load shows substantial seasonal patterns. Figure 13 shows the monthly pattern of the electricity load for each test area. Both test areas show peaks during December, and Kolding hits the minimum during July. Middelfart's minimum monthly load happens during June; however, July is very close to the minimum.

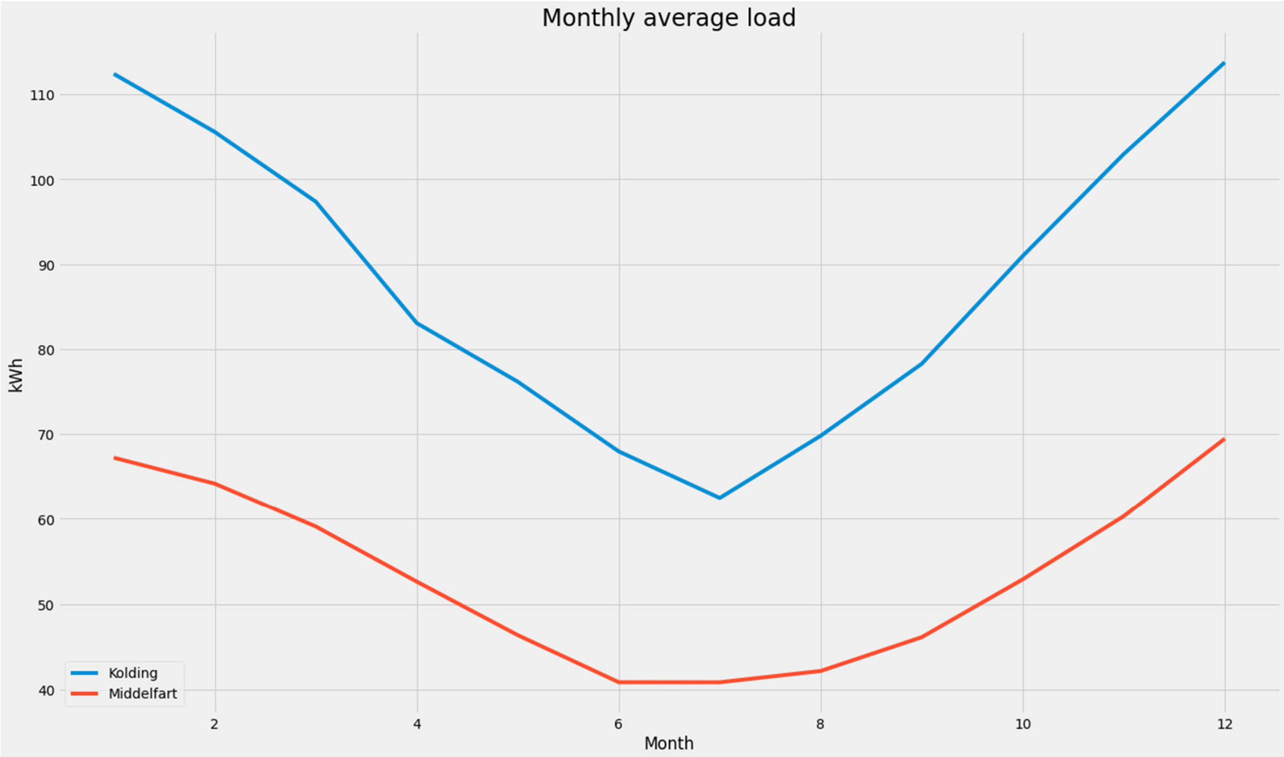


Figure 13 Mean electricity load per month.

Because the monthly variable is cyclic, the sine and cosine transformations are calculated using Equation 5 and Equation 6. The transformation is displayed in Figure 14. Now, December and January are closer than January and March mathematically.

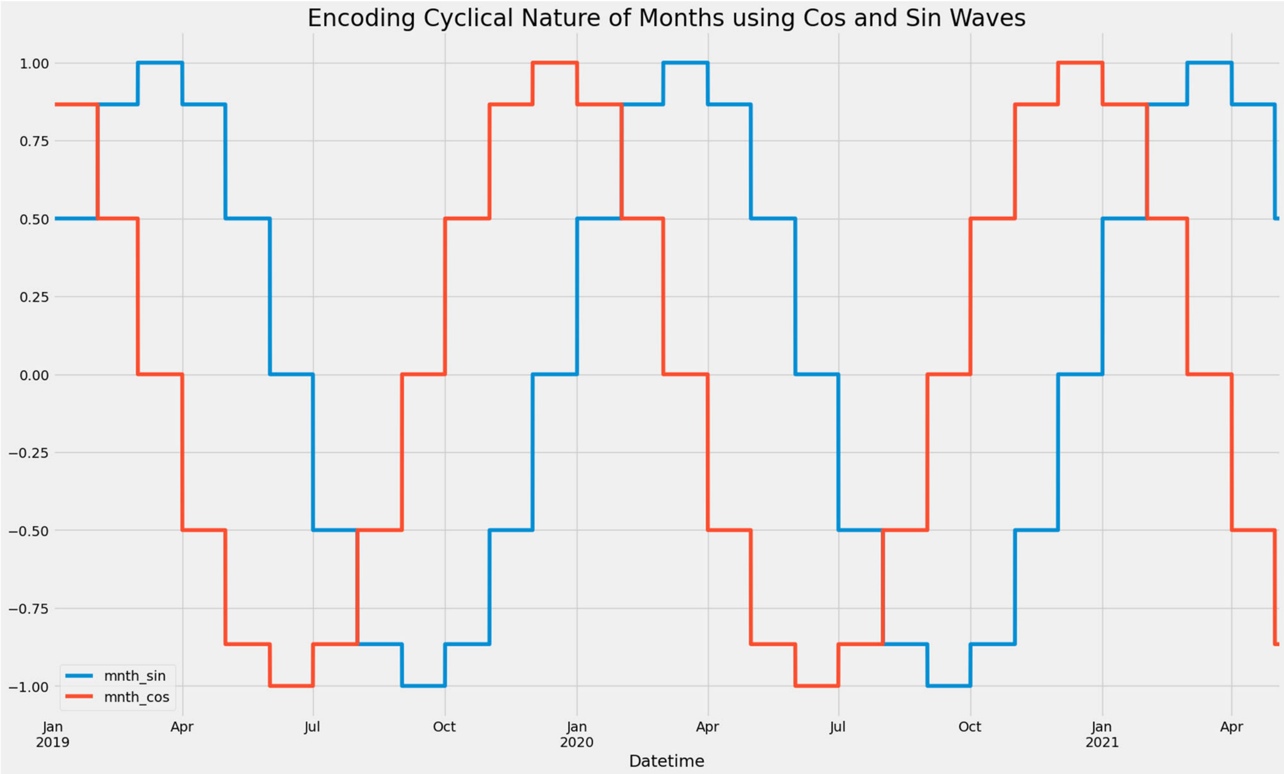


Figure 14 Sine and cosine transformations of month integers.

3.4.6 Summarizing Feature Selection

The previous sections outlined the reasons for selecting the variables. In summary, by removing redundant weather features from the dataset, the risk of a noisy dataset is reduced. Furthermore, the neural network should extract the seasonal and behavioral patterns within the data, as the context of the electricity load has been reinforced with the following variables: sunshine duration, air temperature, day length, weekdays, weekends, and months. Figure 15 shows a heatmap of the Pearson correlation for the load, weather, and day length features. The variables are negatively correlated with the electricity load value, but each contributes with a different context in occupant behavior, seasonality, and time.

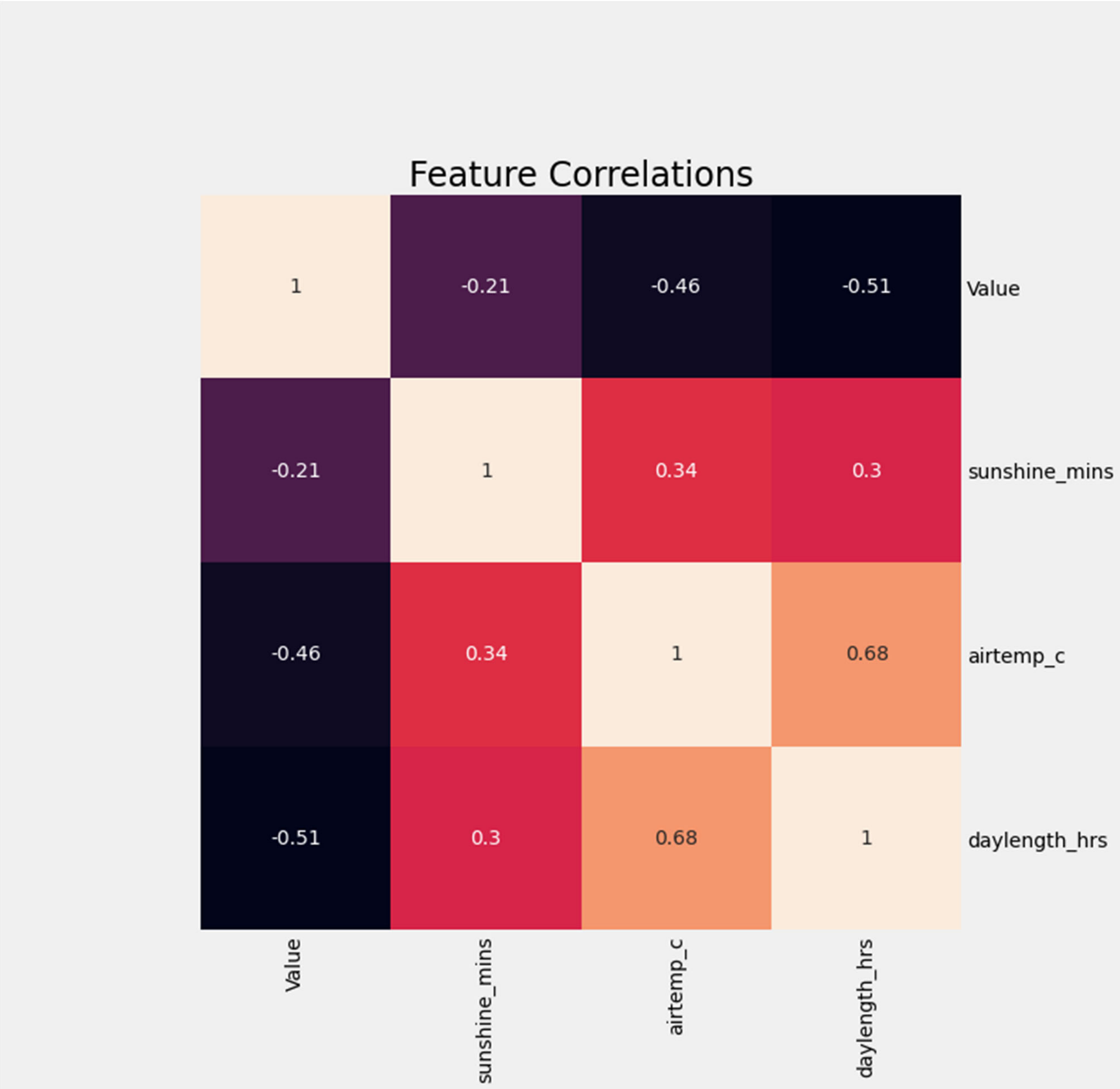


Figure 15 Pearson correlation coefficient of day and time features.

Further, by feature engineering date and time of the load measurements, the weekday, weekend, and month variables are created. The cyclic nature of weekdays and months has been captured by sine and cosine transformations, which turned the integers into two dimensions.

3.4.7 Data Splitting & Inputs

Two data splitting methods were tested to create the inputs for the neural network. Because of the limited amount of data, it can be argued that using a conventional percentage split for training, validation, and test data is not viable. For the conventional split method, the data would be split in an 80% / 10% / 10% ratio. This split would result in just ≈1.87 years of training data, meaning the neural network would not learn a full year of data properly, as it has only seen some timestamps from two years one time. Furthermore, because of the seasonality in the datasets outlined in the previous sections, the model would validate and test specific seasonal patterns in the data, i.e., validation data contains mainly the winter period. In contrast, the test data contains spring data primarily.

Thus, another method is proposed to spread out the training and validation data to cover a full two years of data to combat this. At the same time, the remaining months in 2021 are reserved solely for testing evaluation. This data splitting method uses an offset every 8th day as a validation data point. By leveraging the offset split method, the validation data will still follow a chronological order of Monday to Sunday but are spread apart by a week for every new day. Figure 16 explains where the split happens using an 80/10/10 percentage split compared to the offset split method. The code function for this method can be seen in Appendix C.1. Some disadvantages to the offset method are that data leakage could occur because the days are picked from the training data. Meaning, the days before and after the offset day could somehow be visible in the validation data point. However, because of the data limitations, it is preferred to have some data leakage in the validation set if the model performs much better on a larger test dataset with more diverse patterns and seasonality.

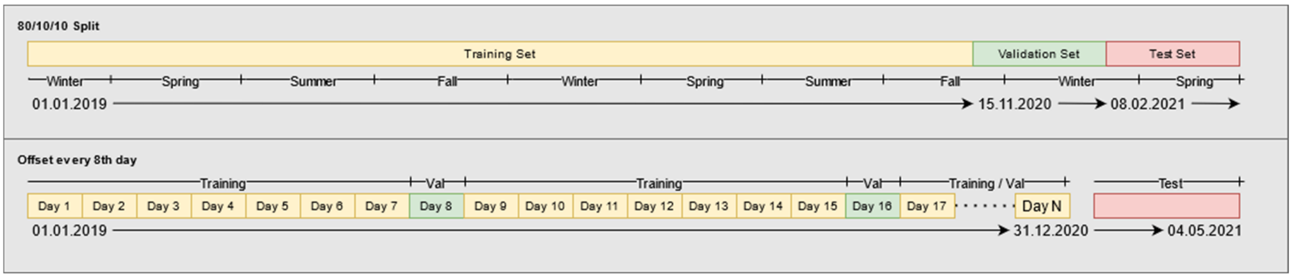


Figure 16 Two different data split methods are illustrated.

3.5. Hyperparameter Search

A hyperparameter search and network architecture testing were conducted using the machine learning tool Weights & Biases *[wandb.ai/site]*. Five rounds of random searches were done to understand better the proposed hybrid neural network and its architecture, totaling around 250 tested models. Each round tested different combinations and ranges of parameters, architectures, datasets, and split methods. The tests were run on a Google Colab GPU (Tesla T4) to speed up the random search sweeps.

3.5.1 Results from hyperparameter sweeps

After testing over 250 different neural networks over five random searches, the following conclusions could be made:

- GRU trained faster and performed somewhat better than LSTM.
- The shallower, simpler models performed much better on test data, meaning one convolutional layer and one gated recurrent unit were enough.
- The best performing optimization algorithm was Stochastic Gradient Descent with momentum (SGD).
- When using SGD, a higher value for momentum increased performance.
- There was generally a difference in performance depending on the dataset used. The Middelfart, Strib dataset was easier to learn for the models than the Nørre Bjert, Kolding dataset.
- Using the conventional 80/10/10 data split resulted in a higher validation loss but a low test loss.
- Using the alternative offset data split resulted in a low validation loss and low test loss.
- Larger batch sizes improved overall performance.
- Models achieved their best validation and test loss after training for a minimum of 25 epochs.
- The more input data the network is allowed to see, the better the performance (max. lookback = 31 days = 744 hours)
- The optimal filter sizes in the convolutional layer and the number of units in the GRU were found by randomly choosing from a log uniform distribution ranging from 32 to 256. The optimal numbers were then rounded to a number divisible by eight to follow conventions.

All runs and experiments with the hyperparameters and configurations can be accessed and explored through the Weights & Biases Dashboard for this thesis found at the following webpage *[wandb.ai/nbvanting/thesis]*.

3.5.2 Grid search sweep

After the random search sweeps, one grid search using the best performing parameters from the random search was conducted. Some parameters and decisions were set as defaults, such as the split method, the optimizer with its hyperparameters, batch size, and dropout rate. The combination of small and large filter and unit sizes for the convolutional and recurrent layers is left open for the grid search. At the same time, kernel sizes for the convolutional layer were left flexible.

3.6. Coding Environment

For this thesis, several frameworks throughout the data pipeline were used. The raw meter data was processed and aggregated in a *Zeppelin Notebook* running on a *Spark Cluster* on the *Center Denmark Data Lake [centerdenmark.com]*. After the initial data processing, the analysis, feature engineering, and external data sourcing were done on a local machine in *Jupyter Notebooks*. For the neural network training, validating, and testing, the *Tensorflow Keras* framework was used and run primarily in *Google Colab Notebooks* on a GPU. Most of the code was written in Python, with some scripts in R. All code for the thesis can be accessed on this GitHub page [github.com/nbvanting/thesis], where each Jupyter Notebook has the corresponding chapter of the thesis in its title. A selection of the code can also be found in Appendix B and Appendix C.

3.7. Proposed Model

This section presents the final proposed hybrid model based on the previous hyperparameter searches and tuning. From Figure 17, we can see the model architecture from the input layer to the final output layer.

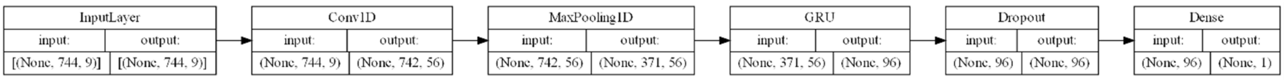


Figure 17 Proposed model architecture.

The input shape is the lookback parameter of 744 hours and the nine variables. The first convolutional one-dimensional layer and ReLU activation function are followed by a pooling layer reducing the size of the features by pooling the maximum values within a window. After the convolutional block, the features are fed into the Gated Recurrent Unit, which outputs 96 units using the default hyperbolic tangent activation function. Then a dropout rate of 0.2 is applied to act as a regularizer, and finally, the output layer returns the predicted electricity load for the next hour. The optimizer algorithm used for the hybrid model is Stochastic Gradient Descent with a momentum of 0.9 and a decaying learning rate starting at 0.04, reducing by a factor of 0.6. SGD with momentum helps the model not to get stuck in local minima by accelerating the gradients in the right direction during gradient descent. Based on much testing, this simple model should be able to perform well on both datasets. Figure 18 visualizes the final hybrid model and its configuration.

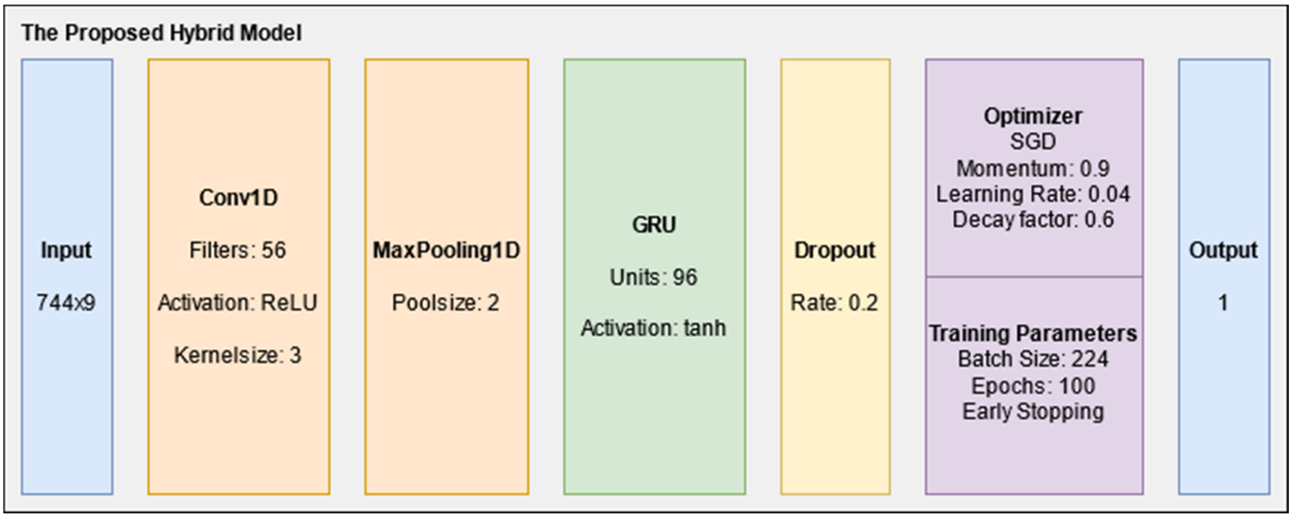


Figure 18 Proposed model configuration.

4. RESULTS

4.1. Baseline Metric

Two different baseline metrics have been calculated to demonstrate if a neural network is a valuable method for predicting real-world electricity load data.

4.1.1 Naïve Approach

The naïve approach uses previous year data to predict the 2021 test set. The training data from 2019, 2020, and an average of the two years will be used for predicting 2021 data. Table 8 shows the performance metrics using the naïve approach on each dataset.

Table 8 Naïve approach results.

Dataset	RMSE (kWh)	MAE (kWh)	MAPE (%)
	Nørre Bjert, Kolding		
2019	22.87	17.68	17.05
2020	20.97	15.95	15.45
'19 & '20 Average	***20.05***	***15.07***	***14.35***
	Strib, Middelfart		
2019	17.35	14.52	21.42
2020	***13.91***	***11.11***	***15.85***
'19 & '20 Average	14.86	12.19	17.64

From Table 8, we can see that the best performances are found using the 2019 and 2020 average load on the Kolding dataset, whereas the Middelfart dataset performs best on just 2020 data. For the final comparison, only the lowest achieved errors will be included.

4.1.2 Commonsense Approach

The second method applies a commonsense approach, which uses the previous observation as a base of prediction. Only test data starting from the 1st of January 2021 will be used for the commonsense approach. Table 9 shows the error metrics on test data for the naïve forecast method.

Table 9 Commonsense approach results.

Dataset	RMSE (kWh)	MAE (kWh)	MAPE (%)
Nørre Bjert, Kolding	33.14	24.59	18.41
Strib, Middelfart	16.88	13.08	15.97

Overall, this approach does not decrease the error compared to using previous year data as the base for predictions. Furthermore, comparing these two baseline approaches shows some varying performance on each dataset, with the Kolding dataset having a much higher error overall. In the coming sections, the differences from each dataset will be explored further.

4.2. Benchmarks

This section shows three different neural networks' performance on the datasets before testing the proposed hybrid model. This benchmarking is done because a basis for comparison is needed to understand if a hybrid network architecture improves the quality of the forecasts on real-world datasets. While the literature suggests that hybrid models are performing well, one of the results from the literature review was that the performance depends on not only the architecture but also the dataset and forecasting horizon. Therefore, benchmarks on a Convolutional Neural Network, a Long Short-Term Memory network, and a Gated Recurrent Unit network have been made on each dataset to ensure a foundation for comparison with the hybrid model. The model parameters and architecture of the benchmark models are taken from the proposed model so that each sub-model of the hybrid model coincides with the benchmark model. From Table 10, the benchmark models' error metrics can be seen.

Table 10 Benchmark neural networks results.

Model	RMSE (kWh)	MAE (kWh)	MAPE (%)
Nørre Bjert, Kolding			
CNN	35.57	26.17	22.44
LSTM	11.9	8.73	8.14
GRU	11.55	8.55	8.17
Strib, Middelfart			
CNN	21.58	16.65	22.16
LSTM	10.32	7.85	10.94
GRU	9.8	7.49	10.54

On its own, the convolutional neural network performs poorly on both datasets, while the LSTM and GRU networks perform almost equally. Moreover, the LSTM and GRU models no longer show substantial differences in error depending on the dataset compared to the baseline metrics in the previous section.

4.3. Proposed Model

The results from the proposed model can now be explored after creating baseline metrics and benchmarks of neural networks in the previous sections. Firstly, this section will outline the performance of the model on each dataset. Secondly, the model's ability to predict during certain events, times and factors will be investigated. Finally, the hybrid model will be compared with the baseline and benchmarks to determine if the proposed hybrid model improves the forecast.

4.3.1 Training & Prediction Phases

For both datasets, all parameters, the neural network architecture, and input variables are equal, except for the historical electricity load for each area. The training and model parameters were set according to the hyperparameter sweep in the experimental setup section. Before training, the data is split using the alternative 8th-day offset split method as described in the Data Splitting & Inputs section, and the values are standardized.

During training, the mean squared error is used as the loss function while also tracking the mean absolute error. The learning rate for the optimizer is reduced by a factor of 0.6 if the model does not improve after five epochs. Additionally, early stopping is used to prevent overfitting after ten epochs. Finally, a rolling checkpoint for the best weights of the model is saved. The number of epochs is set to a higher value as early stopping is included. After training the model, predictions are made on the test data from the 1st of January 2021. However, the predicted values start from the 1st of February due to the 31 days of prior data for the model input. The predicted and true values are then scaled back inversely to their original scale. The performance metrics, root mean square error, mean absolute error, and mean absolute percentage error, are then calculated. The training and prediction phase code can be found in Appendix C.2, Appendix C.3, and Appendix C.4 for the Kolding dataset but is equal for the Middelfart dataset.

4.3.2 Results on Nørre Bjert, Kolding Dataset

From Figure 19, we can see the training and validation loss on the Kolding dataset for a total of 69 epochs. The validation loss started plateauing after around 30 epochs, but the lowest value was achieved after the 59th epoch. There are no signs of overfitting as the loss values follow each other well during training, with only a slight difference between training and validation loss values.

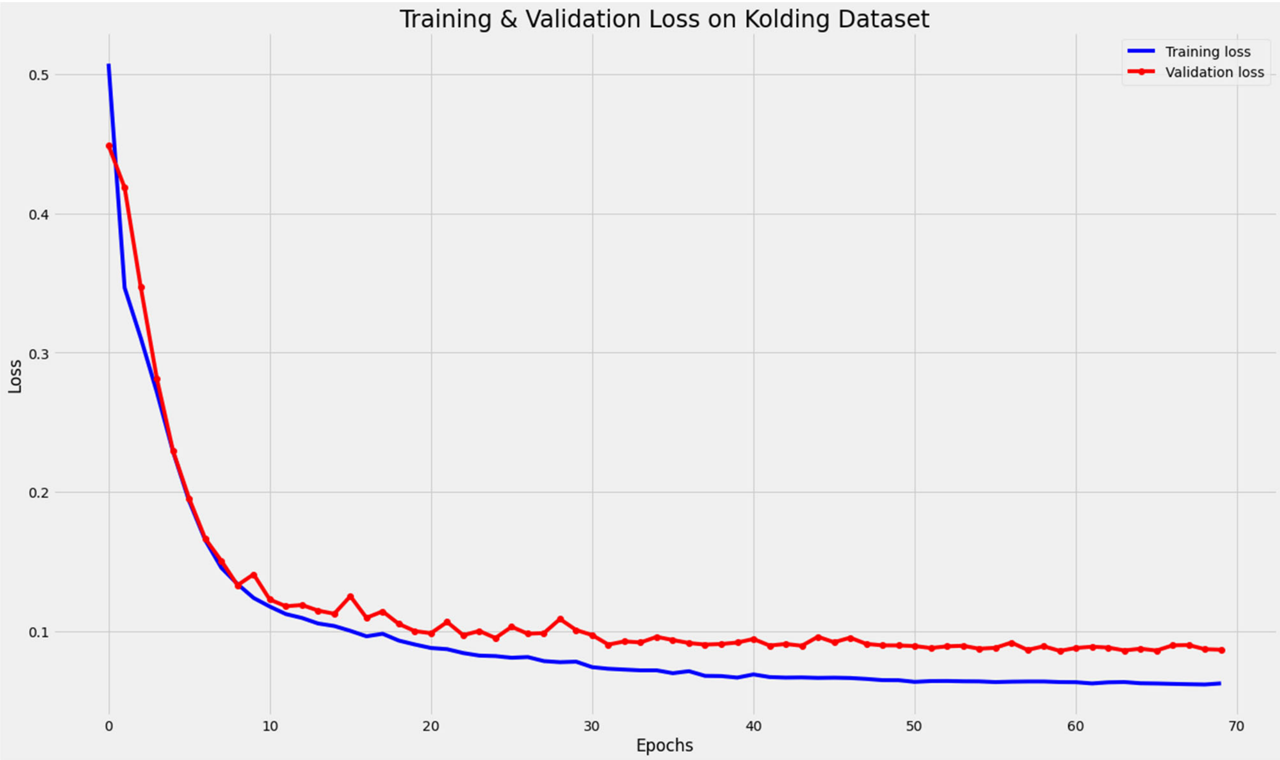


Figure 19 Training and validation loss on the Kolding dataset.

On the test set, the model achieved a slightly higher loss of 0.13 compared to the training loss of 0.06 and validation loss of 0.08. While these values are difficult to interpret, the difference between them can assess the model's ability to predict new data. Considering that the test set only includes data from the first five months of 2021 and the training data includes two years of data with every season of the year, a higher test loss value is expected. After predicting the values for the test set, the following errors are calculated in Table 11.

Table 11 Proposed model's error on the Kolding dataset.

RMSE (kWh)	MAE (kWh)	MAPE (%)
11.11	8.52	8.12

On average, the model misses the true value by 8.52 kWh. There is minimal improvement found when comparing the hybrid model's errors to the benchmark models. Though, there is a substantial improvement from using the naïve and commonsense approach.

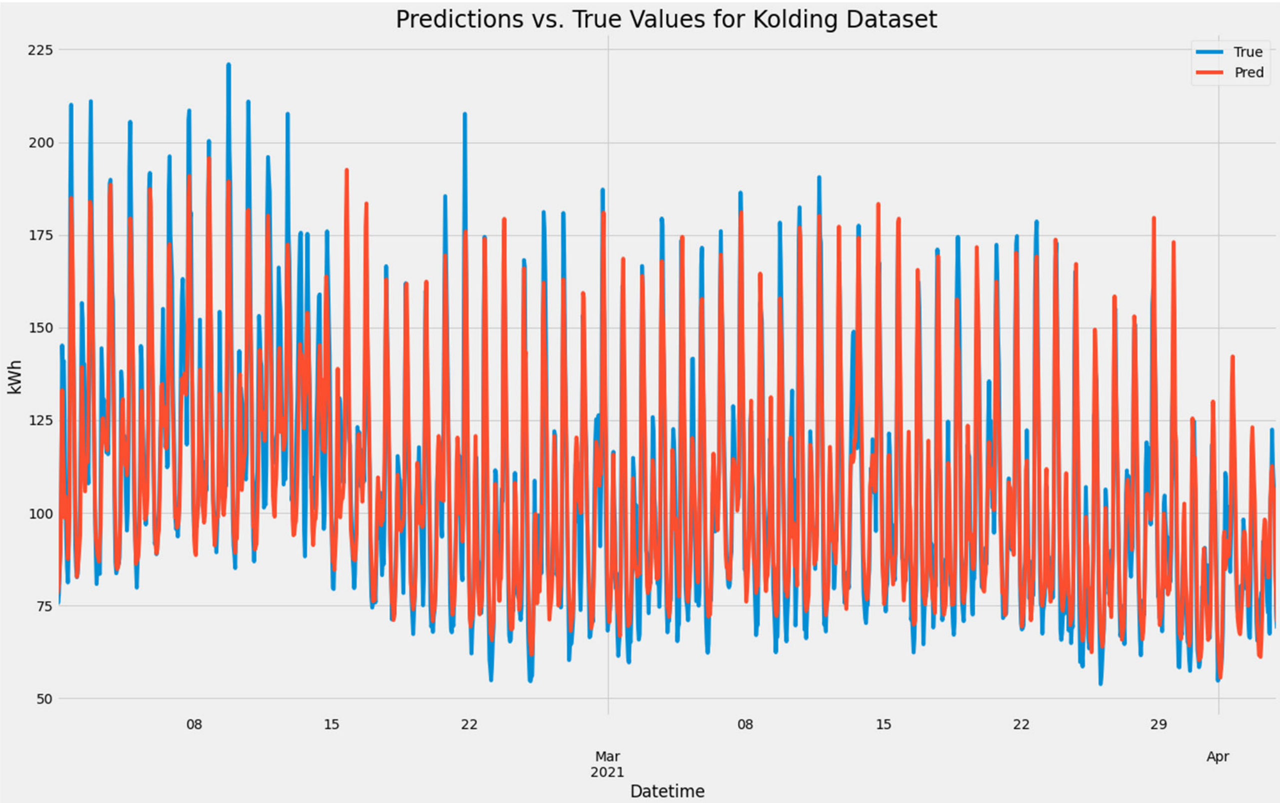


Figure 20 Predicted and true values on the Kolding test set.

From Figure 20, we can see the predictions with the true values of the test data. Overall, the model predicts most of the peaks and valleys of the load and mainly underestimates the peaks. However, the first two weeks show a noisy pattern, further explored in the coming sections. Figure 21 and Figure 22 visualize predictions in the entire month of February and a whole week in March, which should give a better understanding and view of the patterns the model captures.

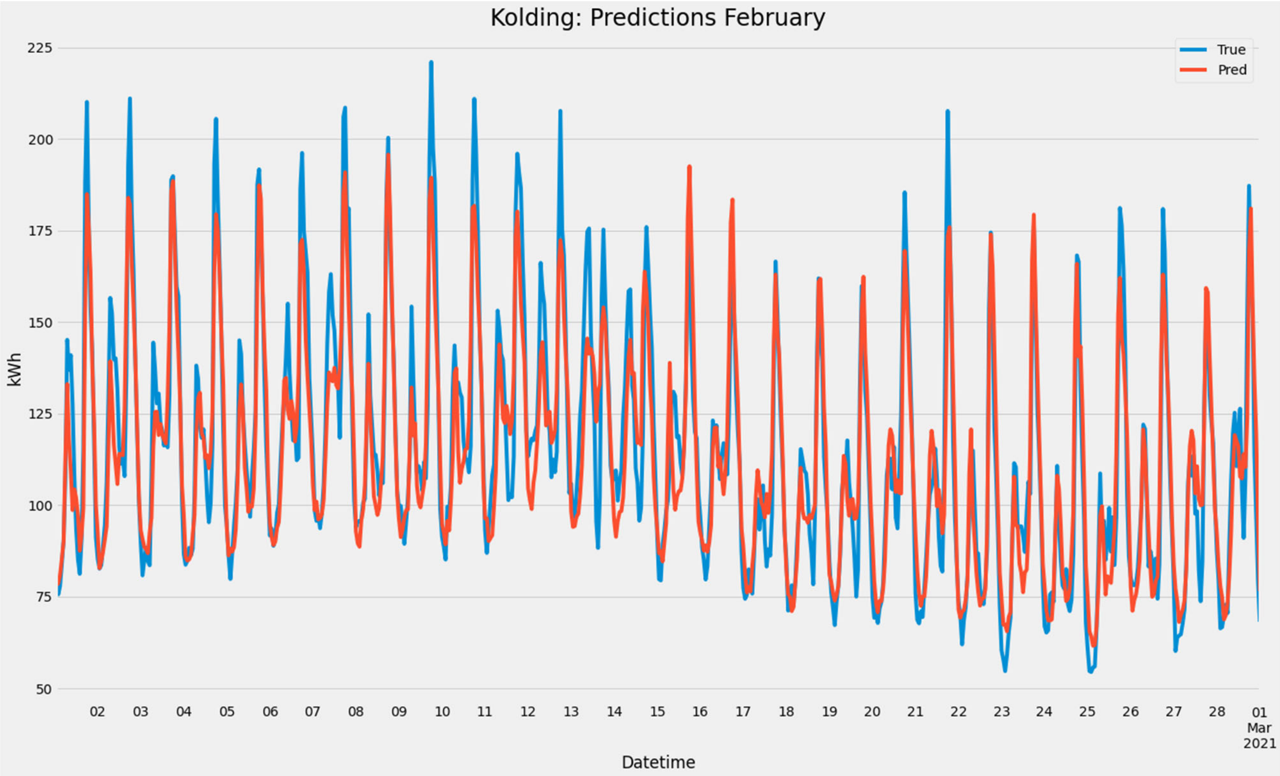


Figure 21 Predictions during February on the Kolding dataset.

From Figure 22, we can see that patterns throughout the day match up well with the true values. Most peaks and valleys are captured correctly, such as the first peak on the 1st of March and the valley after midnight on the 7th. However, the model has difficulties with outlier-looking peaks, such as the first midday peak during the 5th of March, underestimating the peak by around 20 kWh. Similarly, some valleys are challenging for the model to predict when they reach a lower value, observed during the afternoon hours on the 2nd and 4th of March.

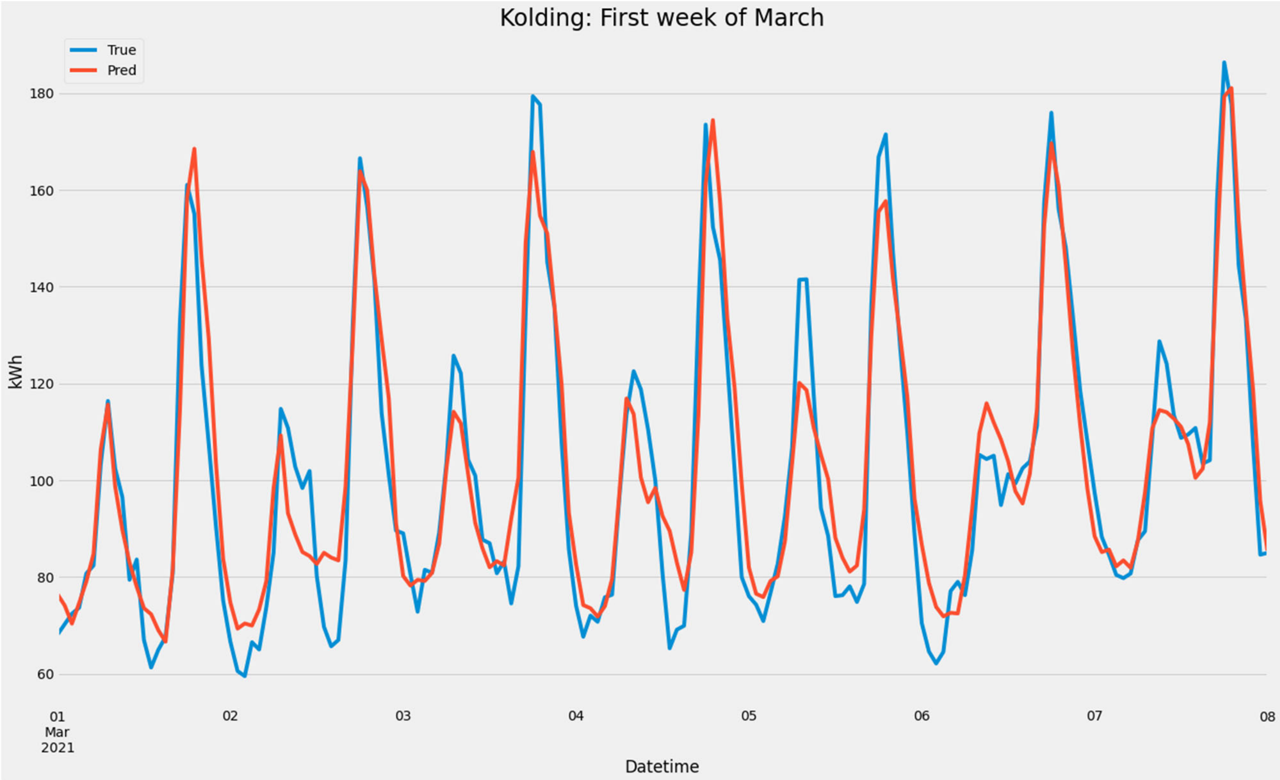


Figure 22 Predictions during the 1st week of March on the Kolding dataset.

In conclusion, the hybrid model performs on par with the LSTM and GRU networks, meaning adding a convolutional layer in front of a GRU has no substantial impact when training and predicting the data from Nørre Bjert, Kolding. The highest errors occur on high peaks and low valleys. However, the model captures the daily two-peak pattern during the morning and evening well.

4.3.3 Results on Strib, Middelfart Dataset

Figure 23 visualizes the training and validation loss for the hybrid model on the Middelfart data. After 48 epochs, the lowest validation loss value of 0.08 and training loss of 0.06 is achieved. A higher test loss of 0.12 on the Middelfart data is expected. Overfitting is not an issue during training, but the model finds no improvement after 58 epochs.

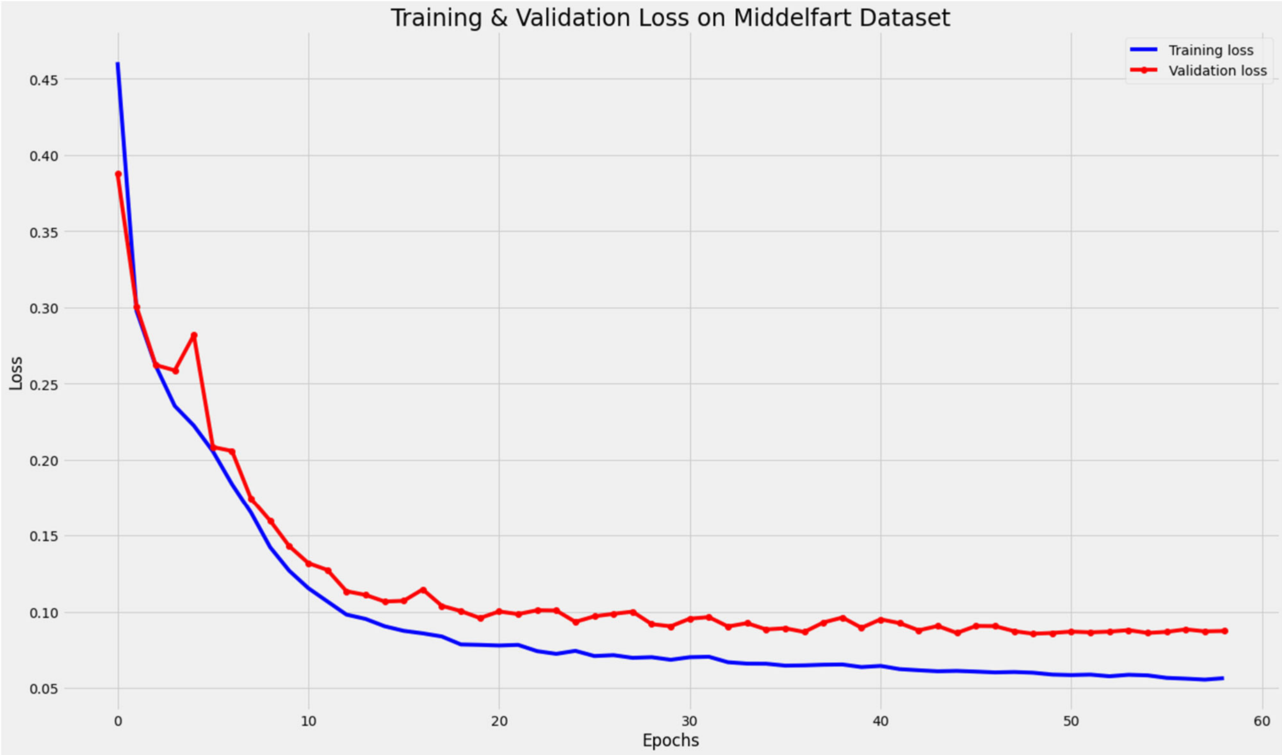


Figure 23 Training and validation loss on the Middelfart dataset.

The errors after predicting the Middelfart test set using the hybrid model can be seen in Table 12. Thus far, these are the lowest errors achieved, and the model misses the true values by just 4.43kWh. Compared to the benchmark neural networks, the hybrid model manages almost to reduce the error by half. For the Middelfart data, there is an improvement to include a convolutional layer in front of a GRU.

Table 12 Proposed model's error on the Middelfart dataset.

RMSE (kWh)	MAE (kWh)	MAPE (%)
5.91	4.43	6.26

The predictions for the Middelfart test data are visualized in Figure X. The model manages to correctly predict most peaks and valleys after the 15th of February. However, the same difficulties as in Figure 20 on the Kolding data can be seen during the first two weeks, further indicating some external factors impacting the electricity load in this period.

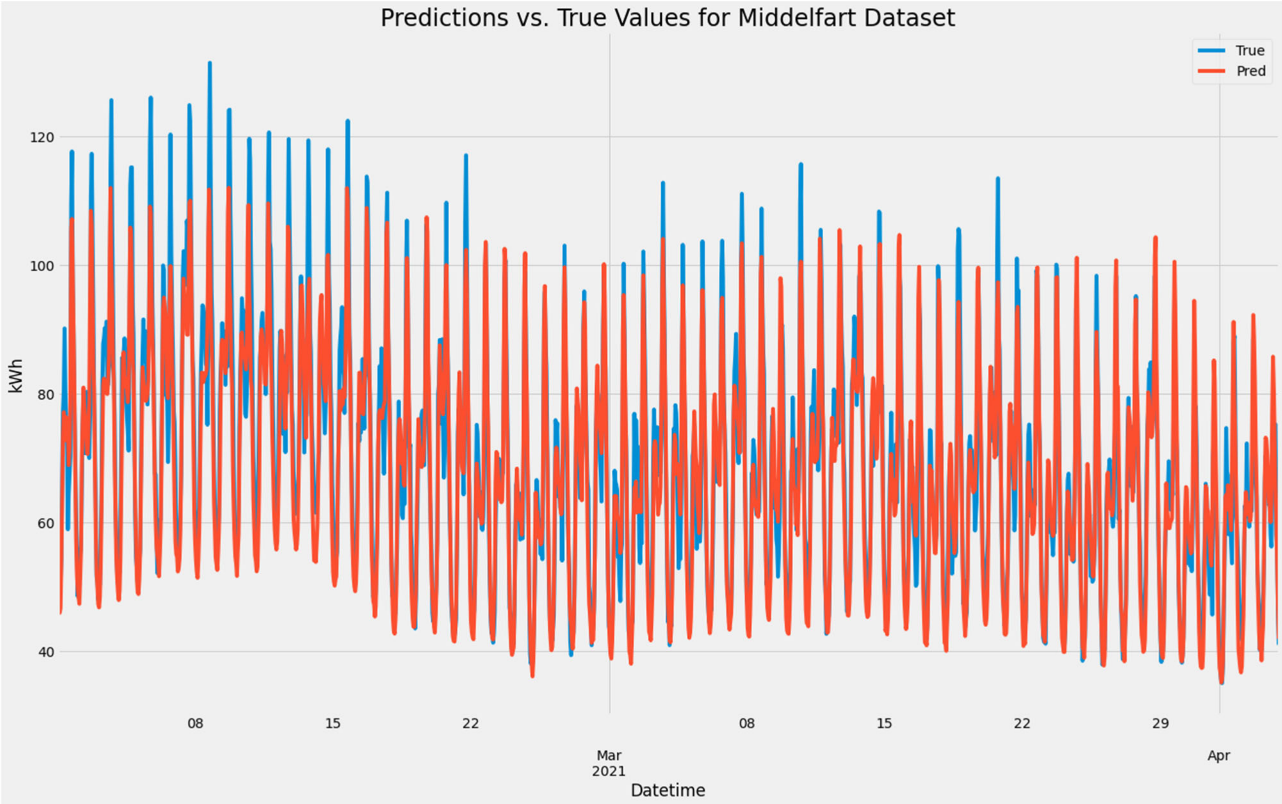


Figure 24 Predicted and true values on the Middelfart test set.

The predictions on the entire month of February and the first week of March are plotted in Figure 25 and Figure 26, respectively. The beginning of February looks to be a difficult period to make predictions for both datasets. However, starting from the 22nd of February, the model captures the true values well.

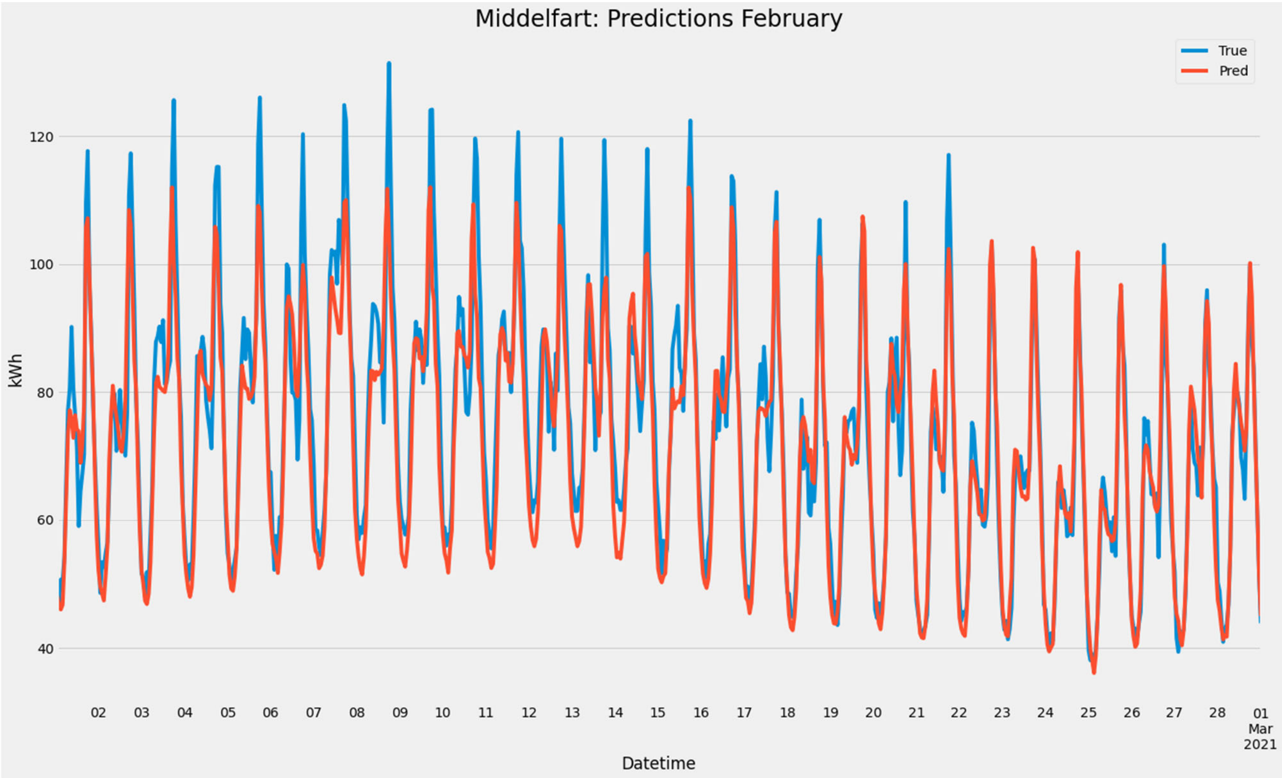


Figure 25 Predictions during February on the Middelfart dataset.

The first week of March shows how close the forecast is to the true load values, especially since the valleys are well-predicted. Though there are still difficulties with the peaks, the model manages to do much better on the Middelfart dataset than on the Kolding data.

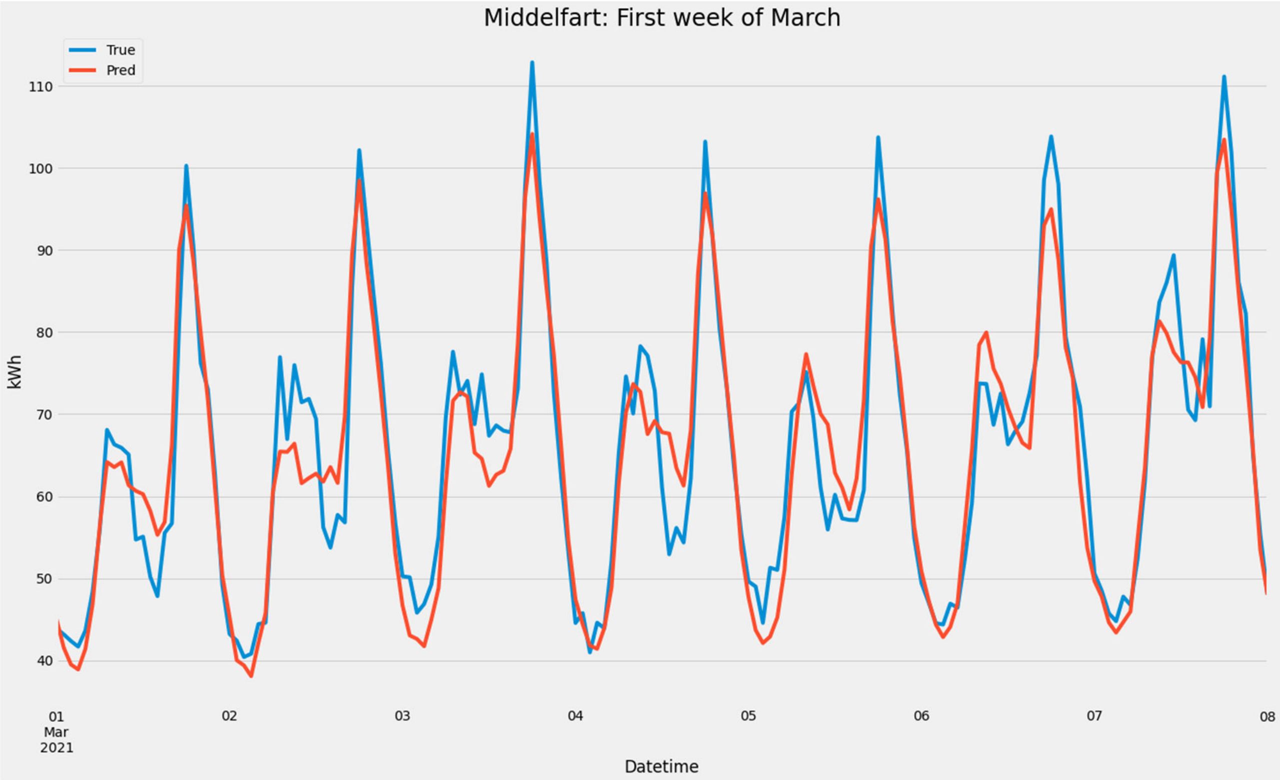


Figure 26 Predictions during the 1st week of March on the Middelfart dataset.

In summary, the proposed hybrid model performs much better than the benchmark models. The addition of a convolutional layer benefits the neural network substantially as it reduces the error by ≈40% on the Middelfart data. Overall, the patterns throughout each day are captured very well, especially the valleys. The most challenging periods to predict are the morning and afternoon, which show slight variations and noisy behavior.

4.3.4 Exploring the Model's Prediction Ability

This section explores certain events and times in the datasets where external factors could impact the electricity load. This exploration is done to understand better the model's decisions and what might influence its capability to do accurate forecasting of the electricity load.

Weather Impact

One such event has shown up in the previous results sections of both datasets as the highly irregular two-week period at the beginning of February. The high peaks and unusual patterns could relate to a freezing period, where temperatures down to almost -10 degrees Celsius were measured.

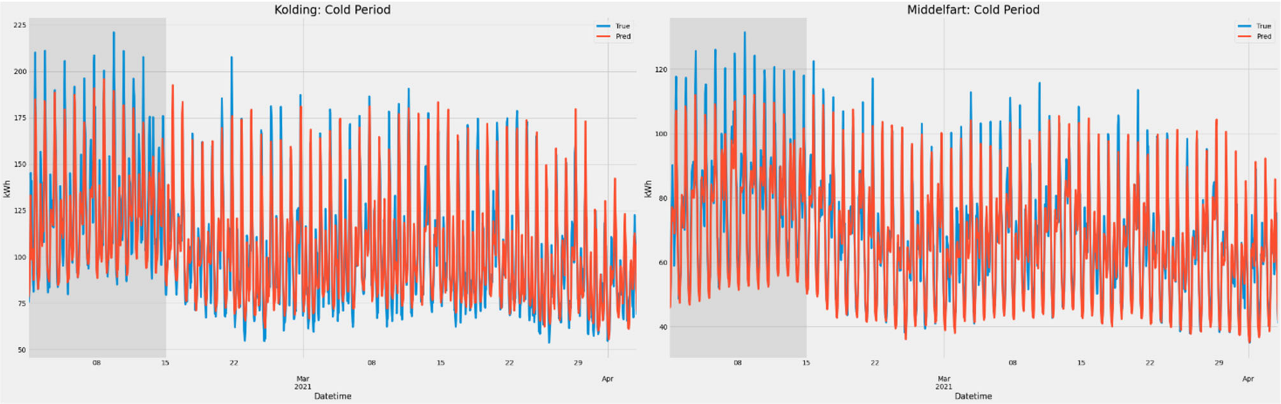


Figure 27 Cold period in the first part of February.

The dark grey area in Figure 27 signifies the cold period, where the electricity consumption increased clearly, compared to the rest of the data points. During this period, the temperature levels are not seen once in the training data, so the model cannot accurately predict. Generally, the model underestimates the high peak loads during this period on both datasets. Following the seasonal and yearly trend, as the temperature decreases, electricity consumption should increase. While the model does capture the higher consumption during the cold period, it overly misjudges how high the peak load should be. This misjudgment can be seen clearly in the Middelfart plot on the right of Figure 27, where the load shifts to a higher value, but the peaks' tips are never reached.

For this reason, there could be another external factor impacting the model during this period. The other weather variable included in the input data is the number of sunshine minutes during an hour. From Figure 28, we can see how both weather variables compare throughout each year. Interestingly, the total sunshine duration during this cold period is higher than the combined duration for the same period in 2019 and 2020. As a reminder, when the sunshine duration increases, the electricity load should decrease slightly based on the trend and seasonality. Thus, the inability to predict the load during this specific weather anomaly could be justified. Because the model has seen only two years of training data, these rare events can be more difficult for the model to learn.

Moreover, the model receives a month indicator feature with the other variables, meaning the model has not seen the combination of this weather in February. In conclusion, to better forecast accurately during such events, the model would need more historical data to learn different weather anomalies. The two weather variables are counteracting each other during this period, leaving the model to find a compromise prediction of the load. An assumption could be that while the temperature seems to increase the mean load, the longer sunshine duration for this period limits the prediction to reach the actual peaks.

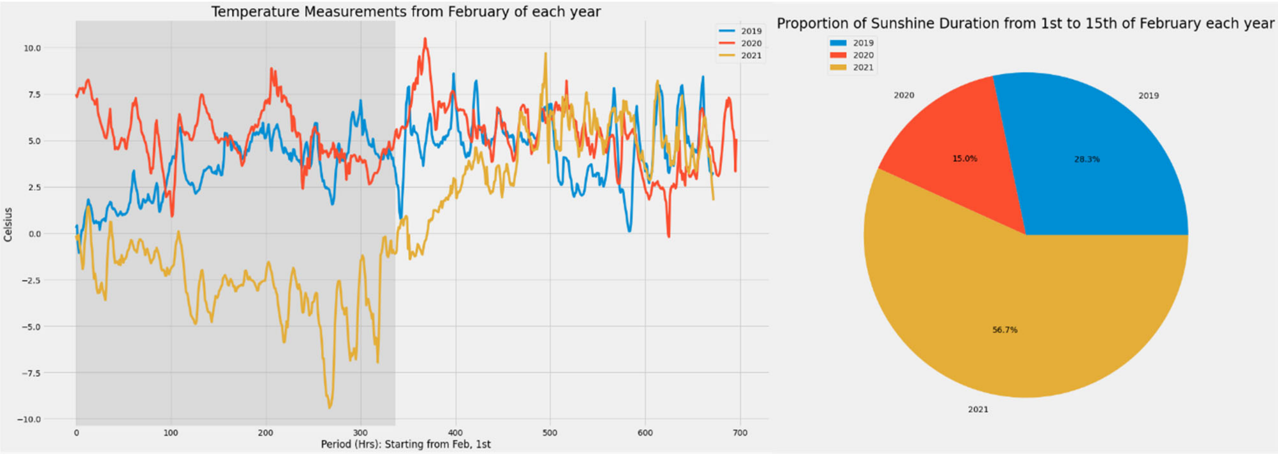


Figure 28 Illustrating different weather during the start of February.

Holidays

One thing that has not been considered for the data input is the holiday indicator variable. For the entire test dataset, there are three holidays and a one-week vacation in week seven. Overall, the model predicts the one-week vacation well but cannot accurately forecast different patterns during religious and national holidays. Figure 29 shows the vacation week seven predictions on each dataset with a mean absolute error of 8.37kWh and 4.18kWh for Kolding and Middelfart, respectively. For both datasets, this is slightly better than the MAE on the entire test set. A few peaks are not correctly predicted, and the midday to afternoon load is slightly irregular, which can be expected during a vacation week.

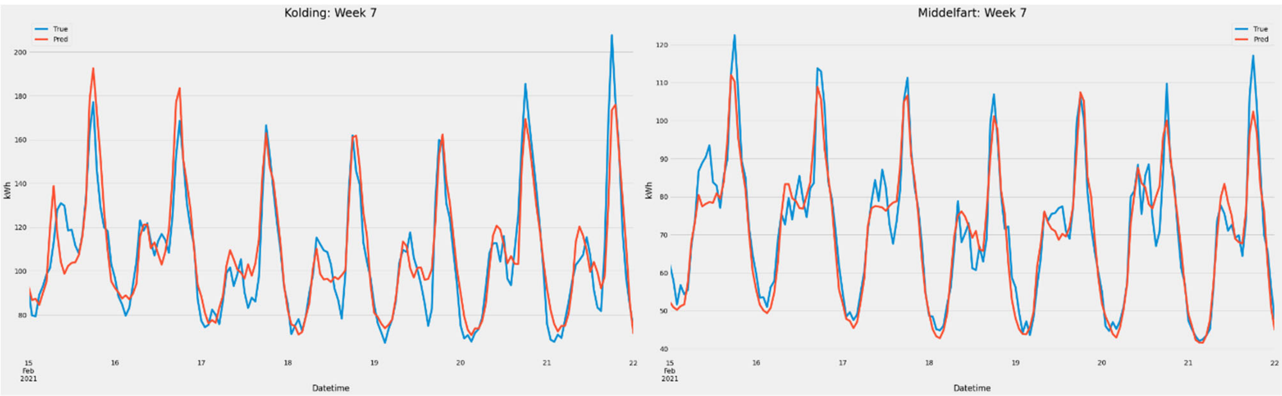


Figure 29 Vacation week seven predictions.

The religious and national holidays occur on the 28th of March (Palm Sunday) and the first and second of April (Maundy Thursday and Good Friday) in the test set. From Figure 30, we can see that the predictions are overestimated during these holidays. It can be argued that Palm Sunday is a different type of holiday compared to Maundy Thursday and Good Friday because it falls on a Sunday. It is interesting to see how the model performs on holidays that fall on weekdays, as the behavior of consumers on weekends changes, as seen in the Date & Time Features section. The characteristic of a holiday is seen clearly in the morning to afternoon period, which shows higher consumption closer to the evening peaks. Further, the peak is lower than during a usual weekday evening, which the model on these holidays overestimates.

From the Kolding predictions on Palm Sunday, we can see that the holiday seems to impact the ability to predict the Monday load accurately. The MAE for the holidays and vacation week on the Kolding data is 8.52 kWh and 4.25 kWh on the Middelfart data, which is the same for Kolding and slightly lower for Middelfart on the entire test set. In summary, the model's error does not look to be impacted heavily by the holidays and vacations.

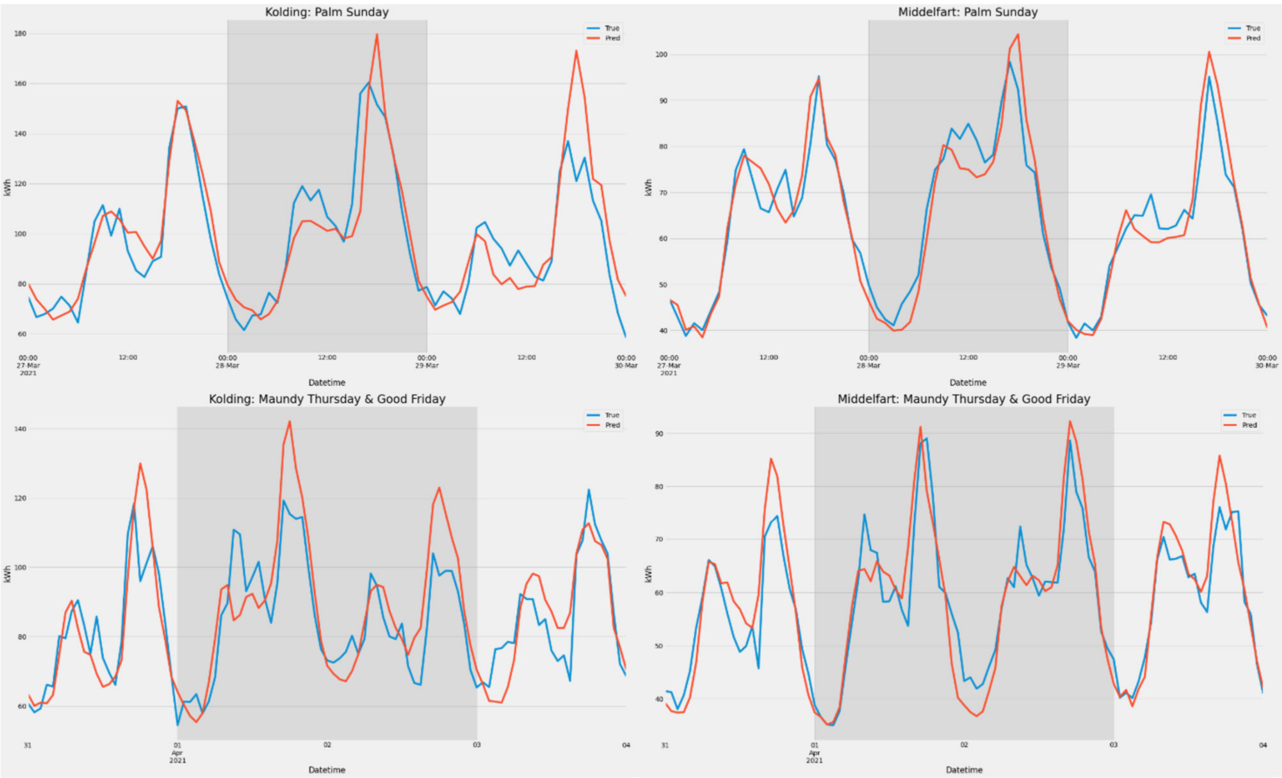


Figure 30 Single holiday predictions.

Day and Time Impact

The following explores the hybrid model's ability to predict during hours of the day, weekdays, weeks, and months. This exploration is done to understand the time variables better and what might influence the predictions. For this section, only the mean absolute error will be compared and analyzed. In Appendix D the plots for each error metric can be found.

Starting with the predictions for each hour of the day, we can see from Figure 31 that the error corresponds to the two-peak pattern found in the daily electricity load. As seen before, this indicates the model's inability to perfectly capture the higher peaks, especially in the evening period. The model reaches its highest error on both datasets during the 18th hour, which is considered the peak hour for electricity consumption due to everyone being in their homes, cooking food, using electrical appliances, and watching television at the same time. Equally, during the morning time, there can be peaks that increase the error. The lowest errors are found after midnight until morning, where the electricity consumption does not vary as much.

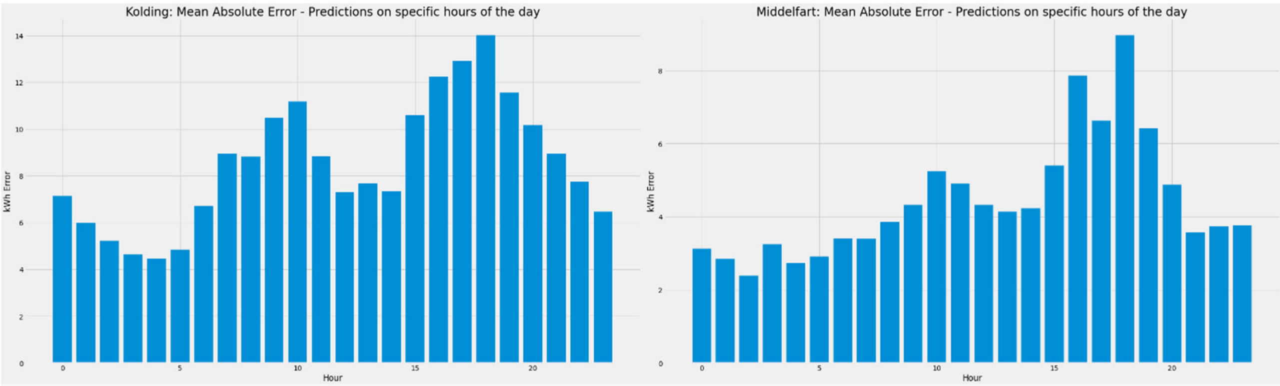


Figure 31 Hourly mean absolute error.

Figure 32 shows the mean absolute error for each weekday. While all weekdays seem to be close to each other in MAE, a few weekdays in each dataset look easier and harder to predict. In the Kolding data, the model predicts worse on Thursdays and better on Fridays. For the Middelfart data, Tuesdays seem to be more challenging to predict. The error is higher on the weekends, which is expected due to the change in occupant behavior.

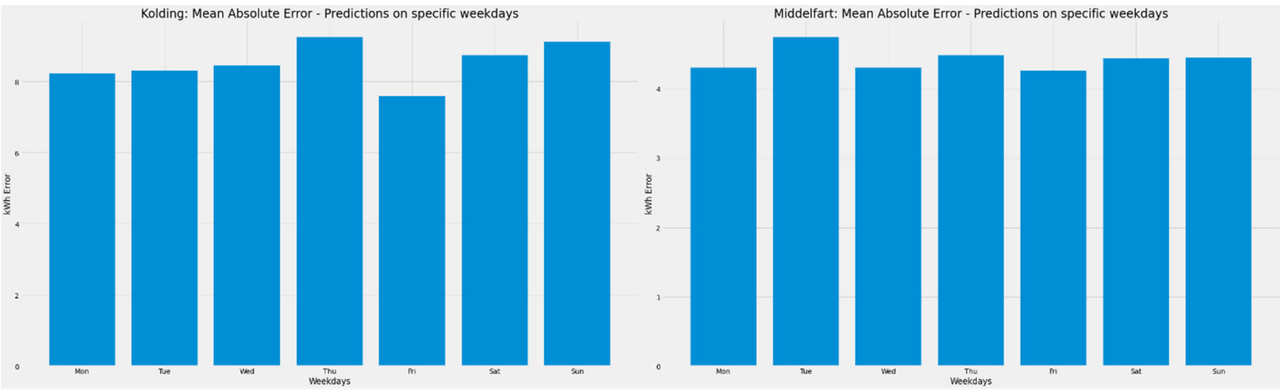


Figure 32 Weekday mean absolute error.

The test data consists of nine weeks ranging from week five to week 13. From Figure 33, we can see the impact of the cold period during weeks five and six. There is a higher error during these two weeks, especially during week six, where the lowest temperature was measured. Consequently, the cold period in week six impacted the overall error rate of the model for both datasets.

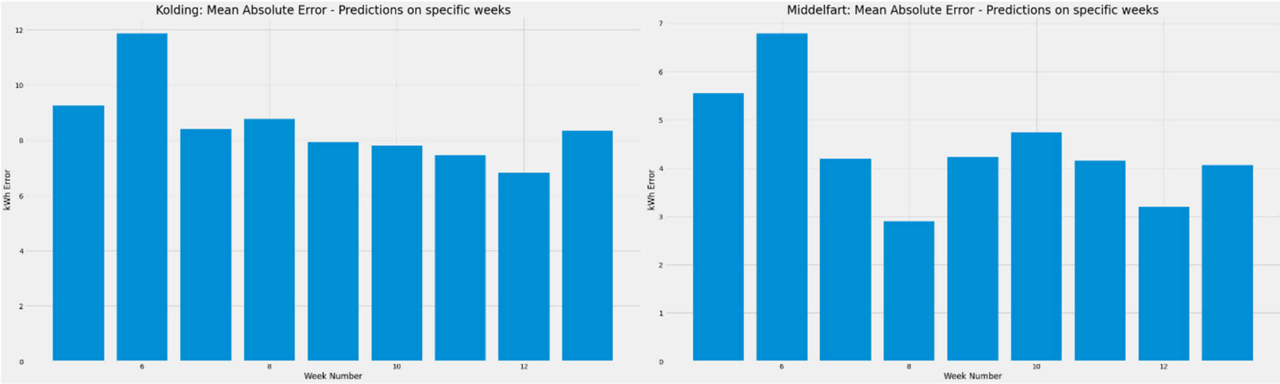


Figure 33 Weekly mean absolute error.

Likewise, the cold period is visible when looking at the monthly error rate from Figure 34. We can see a higher error rate in February compared to March. It must be noted that April only includes the first four days of data due to the shift caused by the lookback parameter and can not be compared to the previous two months.

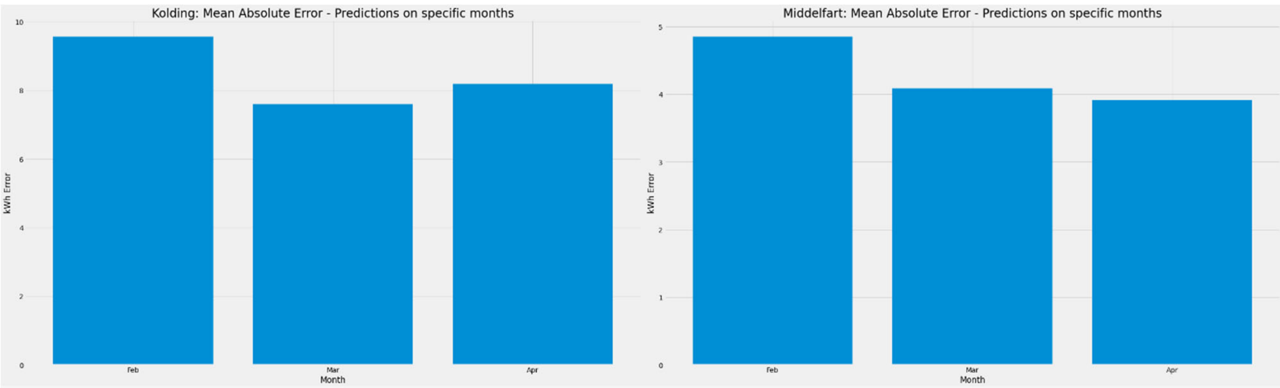


Figure 34 Monthly mean absolute error.

To summarize, the electricity load is mirrored in the errors during certain weekdays and times throughout the test set. It can be argued that more time features could be included as features for the model. The inclusion of hours of the day could help flatten the error throughout the day. Further, it is not easy to see the effect of the month variable on the model, as there are only two whole months of data in the test set. Therefore, it could be beneficial to reduce the lookback parameter from a whole month to one or two weeks to include more predictions. However, as described in the experiments section, the lowest error was found when looking back further. The optimal scenario would be to have more data available, which means this experiment should be repeated when that is the case.

4.3.5 Comparing the Results

The hybrid model performed better when comparing it to the baseline and benchmarks, especially on the Middelfart dataset. After exploring the model's predictions in the previous section, we now understand why there are errors in the predictions and where they mainly occur. Table 13 compiles the error metrics of each approach in an overview.

Table 13 Error comparison table for all methods.

Metric	Naïve Approach	Common-sense Approach	LSTM	GRU	CNN	Proposed: CNN-GRU
Nørre Bjert, Kolding						
RMSE (kWh)	20.05	33.14	11.9	11.55	35.57	***11.11***
MAE (kWh)	15.07	24.59	8.73	8.55	26.17	***8.52***
MAPE (%)	14.35	18.41	8.14	8.17	22.44	***8.12***
Strib, Middelfart						
RMSE (kWh)	13.91	16.88	10.32	9.8	21.58	***5.91***
MAE (kWh)	11.11	13.08	7.85	7.49	16.65	***4.43***
MAPE (%)	15.85	15.97	10.94	10.54	22.16	***6.26***

We can see from the results that the same hybrid model performs differently depending on the dataset. While the error for the Kolding data stays nearly equal compared to the GRU and LSTM networks, the error is reduced by around 40% on the Middelfart data. For the aggregated load in the Kolding residential area, an MAE of 8.52kWh and an MAPE of 8.12% were achieved on the test data. For the Middelfart residential area, the model achieved an MAE of 4.43kWh with an MAPE of 6.26% on the test data. Since everything is equal for the model on both datasets except for the electricity consumption, we can say that 1) the hybrid model has advantages over non-hybrid models, and 2) the composition of the electricity load impacts the performance substantially. Thus, the certainty of the model's performance varies depending on the location, consumer types, and possibly the type of heating that the consumer has installed. As explained in the Feature Selection & Feature Engineering section, the proportion of installations with heat pumps and electric heating is much higher in the Kolding dataset. Heat pumps and electric heating can change the load pattern differently as it is controlled by other factors that are not considered for this experiment.

Thus, this could be one reason for the higher error on the Kolding dataset. There were also signs of overall better performance on the Middelfart data during the hyperparameter search and architecture testing. The non-hybrid CNN model shows the worst overall performance by having a higher error than the two baseline approaches. This result is interesting because it shows the role of the convolutional layers in the hybrid model, where it does not necessarily act as a predictor but as a feature extractor and dimension-reducer for the GRU layer. Figure 35 illustrates the difference between the two benchmark RNNs and the proposed hybrid model. This figure shows that the GRU and LSTM are very similar in their error, and the hybrid model is slightly better on the Kolding dataset. However, on the Middelfart data, the hybrid model outperforms the two RNNs by a large margin, indicating a crucial advantage when applying the hybrid architecture to a neural network.

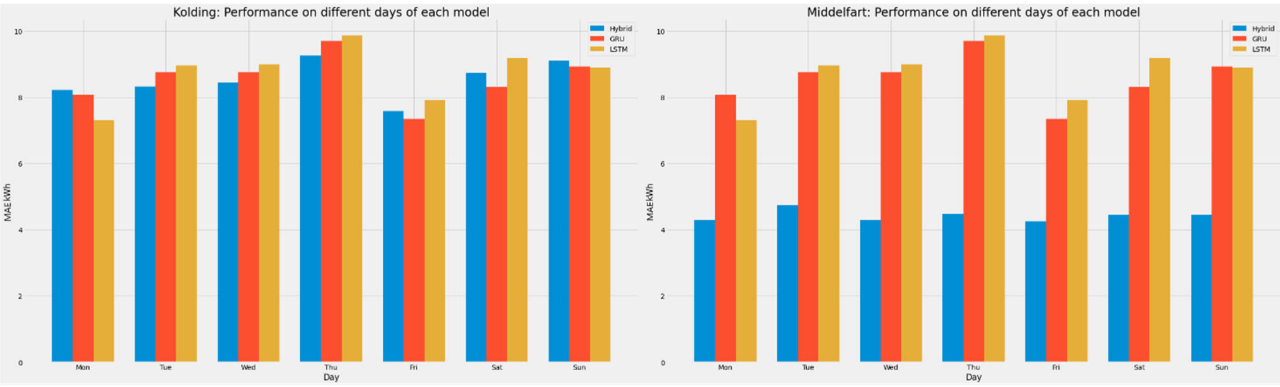


Figure 35 Daily mean absolute error on benchmark and proposed model.

This experiment used a one-hour forecasting horizon with 31 days of previous data on two datasets with different characteristics. As suggested by the literature, a hybrid model consisting of convolutional and recurrent layers improves the forecast of electricity load for neural networks, proved by the model's performance on the Middelfart dataset. At the same time, the model's performance is dependent on the composition of the historical electricity consumption and the forecasting horizon, as seen on the Kolding data.

5. DISCUSSION

This thesis proposed a hybrid neural network to improve the electricity load forecast for two different residential areas in Denmark; Nørre Bjert, Kolding, and Strib, Middelfart. Historical load, temperature, sunshine duration, and time features were used to create multivariate inputs to predict the electricity load for the next hour based on 31 days of previous data. The thesis sought to answer how hybrid neural networks can improve the forecast of electricity load and what advantages and disadvantages there are to this method. The results indicate that hybrid neural networks consisting of convolutional and recurrent layers have advantages over non-hybrid networks. Compared to LSTM and GRU networks, on a short-term horizon of one hour, the proposed model reduced the error by 40% on all performance metrics for the Middelfart data, while no improvement was shown on the Kolding data. Further, the results build on the existing evidence that convolutional layers extract useful information from the data when fed into the recurrent layer. Thus, an important finding in this study was not just achieving a low error on the forecast but the overall architecture and configuration of the hybrid neural network that can be applied to similar datasets.

The performance of the hybrid model varied depending on the dataset, indicating a considerable difference in the characteristics of each residential area. The mean absolute errors achieved were 8.52kWh and 4.43kWh for the Kolding and Middelfart datasets, respectively. Furthermore, the model's ability to predict accurately is highly influenced by the time of day and the weather. Substantial differences in error on the predictions were observed throughout the day and during freezing periods. During peak hours in the evening, the MAE increased considerably to 14kWh for Kolding and 8.96kWh for Middelfart. Firstly, this indicates that more extensive feature engineering should have been considered for the model inputs on the time of day. Secondly, the peaks are highly volatile and lacking a predictable pattern.

The combination of low temperatures and high sunshine duration in February 2021 means that the model predicts data it has not experienced in the training data. As a result, the error increased around 2kWh for both datasets. Generally, it would improve the model's performance by having more historical data available. However, data collection for this experiment was gathered in conjunction with a project that started in 2019. A further argument for more data is the validation of the model. The results of this model are based on test data that only includes the first four months of 2021, meaning we only know the performance of the model on winter/spring data. The results say nothing about the performance during the summer, fall, and early winter.

Furthermore, the predictions cover just three of the four months because of the lookback parameter of 31 days. This drawback was limited by leveraging an offset data splitting method that maximized the amount of test data available while still preserving most of 2019 and 2020 as training data. This method reserved every 8^{th} day in the training data as a validation point, meaning weekdays would continue sequentially with a week apart, and the seasonality remained in the validation data. While this method has drawbacks, such as training data potentially leaking into the validation data, it is preferred over a model that has 1) fewer training data points, 2) non-seasonal validation data, and 3) very little test data for the predictions. Applying this method is necessary due to the limited data available. However, the conventional percentage-ratio split, e.g., 80%/10%/10%, could be used after acquiring a larger dataset. Another concern about the limited data was that the test set included a weather-outlier period at the beginning of February that is not seen in the training data once. As seen from the results, this impacted the error metrics negatively, leading to a weaker forecast. The error of March, which contains no outliers, vacations, or holidays, reflects the model's prediction capability, while the February error illustrates its weaknesses. The architectural and parameter decisions of the hybrid neural network are based on an iterative process that developed throughout the experimental phase. Early in the experiments, it was clear from training speed and convergence that the Gated Recurrent Unit was preferable. However, many of the related works focused primarily on using Long Short-Term Memory for the recurrent layer of the hybrid model, as seen in Table 4. For this study, the decision to use GRU came after testing both combinations for runtime and validation loss performance. Hence, this study shows that GRU performs well on load forecasting applications, which was missing in the related works.

Moreover, the hyperparameter and network tuning were done exclusively on the hybrid model, meaning the benchmark neural networks are subsets of the hybrid model and are thus not tuned further. The only change made was to the CNN, which needed an output layer to predict the load, as it served as the first layer of the hybrid model. Therefore, it must be considered that the benchmarks could be tuned and improved further. However, the objective of this experiment was to investigate if the addition of a convolutional layer to the recurrent neural network would increase performance.

As suggested in the literature review, the results show that a hybrid model has advantages over non-hybrid neural network architectures. In line with the literature review findings, there is an evident change in performance depending on the data inputs. One dataset showed considerable improvement from non-hybrid recurrent neural networks, while the other showed none. While the related works often used weather inputs for the neural network, the results from this study can challenge this decision by investigating specific periods with weather anomalies. As seen in the results chapter, the two weather variables seemed to counteract each other, leading the model to predict a higher error during this period. Thus, it could be beneficial to reduce weather parameters further with the limited data available. The experimental results of this thesis show important evidence of leveraging hybrid models for electricity load forecasting. The results of a controlled test were needed in the literature, as there was no solid ground for comparing the models of the related works, as they used very different datasets and forecast horizons. The literature review showed evidence for limited generalizability of the methods. This experiment documented each step of the data pipeline and model training to pinpoint where the difference in performance occurs and how the neural network architectures genuinely compare.

While the difference in data source and configurations of parameters impact the model's error, an unexpected result from this experiment was that two residential areas within 15km proximity showed such a difference in performance. During the initial analysis of the data, it was noted that an installation in Kolding consumes 0.15kWh more per hour on average. While this number did raise some concern, it was not expected to lead to a worse performance later. It did not necessarily imply a different behavioral pattern but rather a higher overall consumption.

This impact raises questions about further considerations before aggregating the load on a substation level. Consequently, it would be necessary to create profiles of each installation to truly understand the composition of the data inputs, which is beyond the scope of this thesis. Some researchers applied this methodology in the related works with success (Aurangzeb et al., 2021; Jarábek et al., 2018). However, an exploration into the installations of each dataset showed that one contained noticeably more electric-based heating installations, which incidentally was the one that performed with a higher error. This behavior of the model indicates weakness to currently unknown aspects, as heat pumps are controlled by factors not considered directly for this experiment, which instead based its feature selection on the occupant behavior and seasonality. Moreover, it can be argued that an introduction of other electric-based objects to the grid on a larger scale, such as electric vehicles, would be reflected in the predictions and expectedly increase the error further.

Effectively, this would make a lot of the historical data unusable if it does not contain consumption from heat pumps or EVs where neural networks can learn these distinct patterns. As electrification and demand increase, it is becoming more and more valuable to collect electricity meter readings to develop this field further and ensure model robustness that includes this consideration.

The uncertainty of the model's performance based on unknown factors is a concern. The model's performance ranges from an MAE of 4.43kWh to 8.52kWh, depending on the constellation of installations on the grid. This hourly range of error can limit DSOs to rely on the model's predictions for larger-scale flexibility applications. However, it is beyond the scope of this thesis to go into a deeper analysis of each installation's impact on the aggregated load, and the change in performance is an assessment based on a general exploration of the highest consumers in each dataset. Before starting the experiments, it was not expected to see such substantial differences in performance on each dataset, so the impact of heat pumps was not considered in the feature selection process.

A limitation of this thesis was the historical data availability. With only two years of training data, one of which could be impacted by nationwide lockdowns due to the COVID-19 situation, the model has learned just about the minimum of an entire year possible. An unconventional method for splitting data into training, validation, and test sets was needed to exploit the available data fully. While this method raises some concerns regarding data leakage in the validation data, it was necessary for this specific experiment. Additionally, with just a few months of test data, the results do not say anything about the error rate for an entire year. From the results, we cannot determine how the model truly performs during the summer vacation, warmer weather, or other holidays such as Christmas and New Year. Predictions during these periods may raise new questions the same way the effect of heat pumps on the model did.

The results from this thesis support the findings in the literature review regarding comparison and generalizability of the neural networks' by underlining the need for controlled tests as conducted in this study. The results are based on data from two residential areas in proximity, meaning they cannot be generalized to areas with different characteristics, such as city centers or industrial areas. Furthermore, the installations are a subset of a larger grid, meaning the aggregated load of a whole grid may not result in the same performance as these 164 and 136 addresses. Finally, the findings of this thesis should mainly be considered for short-term applications for residential areas. Long-term oriented focuses like the future grid state or capacity may need other considerations during the configuration and feature selection, mainly because the model showed weaknesses towards non-occupant electricity consumption, i.e., heat pumps.

6. CONCLUSION

This thesis studied how neural networks can improve the forecasting of the electricity load in danish residential areas and the advantages or disadvantages of this method. A systematic literature review of related works was conducted to determine the state-of-the-art deep learning methodologies for load forecasting. It was found that hybrid neural networks consisting of convolutional and recurrent layers showed advantages over non-hybrid ANNs. Furthermore, the literature review gave no solid basis for comparing and generalizing deep neural networks for Danish residential areas. Consequently, to compare the performance of various configurations of neural networks and determining the advantages, a controlled test needed to be conducted. Based on these findings, this study experimented using the aggregated consumption from two residential areas in Denmark to test the performance of such a hybrid model. The two test areas are located in Nørre Bjert, Kolding and Strib, Middelfart, and included data from January 1st, 2019, to April 4th, 2021. The proposed neural network was compared to baseline metrics and CNN, GRU, and LSTM benchmark ANNs. The experiments reduced error by 40% on all metrics compared to the best benchmark results when predicting the Strib, Middelfart residential area. No substantial improvement was found on the Nørre Bjert, Kolding data compared with the GRU and LSTM networks. The proposed model achieved an MAE of 4.43kWh for the Middelfart test area and an MAE of 8.52kWh for the Kolding test area when predicting the next hour's consumption. Thus, by leveraging a hybrid neural network, electricity load forecasting can be improved depending on the area. The disparity in error on the two datasets seemed to originate from different consumption patterns in the data relating to a higher proportion of heat pumps in the Kolding area. This unexpected result exposed a weakness in the proposed model, which indicates a lack of robustness with the addition of unknown factors to the grid. The literature review also found that the advantages of leveraging hybrid neural networks relate to the convolution operation that acts as a feature extractor for the subsequent recurrent layer. The results from the experiments show evidence for this on one of the two datasets, as a considerable improvement was found when using a convolutional block to feed data into the gated recurrent unit. In addition, when analyzing the model's ability to predict based on day and time features, it was found that more extreme weather has a substantial negative impact on the proposed model's forecasting error. During a period of cold temperatures with long durations of sunshine, the model could not predict the evening peak loads.

Based on the findings, it is recommended to leverage hybrid neural networks consisting of convolutional and recurrent layers to achieve the lowest error on Danish residential areas' electricity consumption forecasts. While the model did not perform equally on both datasets, there is a potential error reduction of 40% by applying this architecture. As a result of this thesis, it is also recommended to expand real-world test areas to include more diverse data patterns that can be generalized. More diverse data would determine if the model can deal with the volatility of aggregated load data caused by non-occupant consumption patterns such as heat pumps or EVs. Moreover, it is recommended to repeat this experiment when more data is available, and an analysis of the composition of a grid has been made. To ensure that the model's representability, data of a minimum of four years should be collected to reserve enough data for two years of training, one year of validation, and one year of testing. This study focused on applying deep learning methodologies to residential areas; however, many more applications are considered with varying performances in the literature. Thus, future work can conduct a similar controlled test using a hybrid neural network on forecasts for non-residential areas, as indicated in the related works section. At the same time, the hybrid architecture should also be tested for medium- and long-term forecasting horizons since the study did not consider this.

7. REFERENCES

Ai, S., Chakravorty, A., & Rong, C. (2020) Household Energy Consumption Prediction Using Evolutionary Ensemble Neural Network. In. *Lecture Notes in Mechanical Engineering* (pp. 923-931).

Al Khafaf, N., Jalili, M., & Sokolowski, P. (2019). Application of deep learning long short-term memory in energy demand forecasting. In *Communications in Computer and Information Science* (Vol. 1000, pp. 31-42).

Amin, P., Cherkasova, L., Aitken, R., & Kache, V. (2019, 8-13 July 2019). *Automating Energy Demand Modeling and Forecasting Using Smart Meter Data.* Paper presented at the 2019 IEEE International Congress on Internet of Things (ICIOT).

Aurangzeb, K., & Alhussein, M. (2020, 10-10 Feb. 2020). *Deep learning framework for short term power load forecasting, a case study of individual household energy customer.* Paper presented at the 2019 International Conference on Advances in the Emerging Computing Technologies (AECT).

Aurangzeb, K., Alhussein, M., Javed, K., & Haider, S. I. (2021). A Pyramid-CNN based deep learning model for power load forecasting of similar-profile energy customers based on clustering. *IEEE Access*. doi:10.1109/ACCESS.2021.3053069

Barzola-Monteses, J., Espinoza-Andaluz, M., Mite-León, M., & Flores-Morán, M. (2020, 16-20 Nov. 2020). *Energy Consumption of a Building by using Long Short-Term Memory Network: A Forecasting Study.* Paper presented at the 2020 39th International Conference of the Chilean Computer Science Society (SCCC).

Bedi, G., Venayagamoorthy, G. K., & Singh, R. (2020). Development of an IoT-Driven Building Environment for Prediction of Electric Energy Consumption. *IEEE Internet of Things Journal, 7*(6), 4912-4921. doi:10.1109/JIOT.2020.2975847

Billanes, J. D., Ma, Z., & Jørgensen, B. N. (2017). Consumer Central Energy Flexibility in Office Buildings. *Journal of Energy and Power Engineering, 2017*(11), 621-630. doi:10.17265/1934-8975/2017.10.001

Bot, K., Ruano, A., & da Graça Ruano, M. (2020). Forecasting Electricity Consumption in Residential Buildings for Home Energy Management Systems. In *Communications in Computer and Information Science* (Vol. 1237 CCIS, pp. 313-326).

Bruno, S., Dellino, G., Scala, M. L., & Meloni, C. (2018, 12-15 June 2018). *A Microforecasting Module for Energy Consumption in Smart Grids.* Paper presented at the 2018 IEEE International Conference on Environment and Electrical Engineering and 2018 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe).

Chan, S., Oktavianti, I., & Puspita, V. (2019, 17-19 Oct. 2019). *A Deep Learning CNN and AI-Tuned SVM for Electricity Consumption Forecasting: Multivariate Time Series Data.* Paper presented at the 2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON).

Chenglei, H., Kangji, L., Guohai, L., & Lei, P. (2015, 28-30 July 2015). *Forecasting building energy consumption based on hybrid PSO-ANN prediction model.* Paper presented at the 2015 34th Chinese Control Conference (CCC).

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. J. a. p. a. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation.

Chollet, F. (2018). *Deep learning with Python* (Vol. 361): Manning New York.

Christensen, K., Ma, Z., & Jørgensen, B. N. (2021). Technical, Economic, Social and Regulatory Feasibility Evaluation of Dynamic Distribution Tariff Designs. *Energies, 14*(10), 2860.

Clark, W. G., & Kelly, E. P. (2016). Demand-Side Management. In *Energy Management and Conservation Handbook*: CRC Press.

Dudek, G., Pełka, P., & Smyl, S. (2020) 3ETS+RD-LSTM: A New Hybrid Model for Electrical Energy Consumption Forecasting. In*: Vol. 12534 LNCS. Lecture Notes in Computer*

Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (pp. 519-531).
Eseye, A. T., Lehtonen, M., Tukia, T., Uimonen, S., & Millar, R. J. (2019). Machine Learning Based Integrated Feature Selection Approach for Improved Electricity Demand Forecasting in Decentralized Energy Systems. *IEEE Access, 7*, 91463-91475. doi:10.1109/ACCESS.2019.2924685
Ferlito, S., Atrigna, M., Graditi, G., Vito, S. D., Salvato, M., Buonanno, A., & Francia, G. D. (2015, 3-5 Feb. 2015). *Predictive models for building's energy consumption: An Artificial Neural Network (ANN) approach.* Paper presented at the 2015 XVIII AISEM Annual Conference.
Flexible-Energy-Denmark. (2019). About the FED project. Retrieved from https://www.flexibleenergydenmark.dk/om-fed/
Gao, Y., Liu, X., Li, X., Gu, L., Cui, J., & Yang, X. (2018, 25-27 May 2018). *A Prediction Approach on Energy Consumption for Public Buildings Using Mind Evolutionary Algorithm and BP Neural Network.* Paper presented at the 2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS).
Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*: O'Reilly Media.
Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*: MIT Press.
Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation, 9*(8), 1735-1780. doi:10.1162/neco.1997.9.8.1735 %J Neural Computation
Hrnjica, B., & Mehr, A. D. (2020) Energy demand forecasting using deep learning. In*. EAI/Springer Innovations in Communication and Computing* (pp. 71-104).
Hubel, D. H., & Wiesel, T. N. J. T. J. o. p. (1968). Receptive fields and functional architecture of monkey striate cortex. *195*(1), 215-243.
Jarábek, T., Laurinec, P., & Lucká, M. (2018). *Energy load forecast using S2S deep neural networks with k-Shape clustering.* Paper presented at the 2017 IEEE 14th International Scientific Conference on Informatics, INFORMATICS 2017 - Proceedings.
Katsatos, A. L., & Moustris, K. P. (2019). *Application of artificial neuron networks as energy consumption forecasting tool in the building of regulatory authority of energy, Athens, Greece.* Paper presented at the Energy Procedia.
Kaur, D., Kumar, R., Kumar, N., & Guizani, M. (2019, 9-13 Dec. 2019). *Smart Grid Energy Management Using RNN-LSTM: A Deep Learning-Based Approach.* Paper presented at the 2019 IEEE Global Communications Conference (GLOBECOM).
Khan, Z. A., & Jayaweera, D. (2018, 9-10 April 2018). *Approach for forecasting smart customer demand with significant energy demand variability.* Paper presented at the 2018 1st International Conference on Power, Energy and Smart Grid (ICPESG).
Kim, T., & Cho, S. (2019, 10-13 June 2019). *Particle Swarm Optimization-based CNN-LSTM Networks for Forecasting Energy Consumption.* Paper presented at the 2019 IEEE Congress on Evolutionary Computation (CEC).
Kim, T. Y., & Cho, S. B. (2019). Predicting residential energy consumption using CNN-LSTM neural networks. *Energy, 182*, 72-81. doi:10.1016/j.energy.2019.05.230
Koprinska, I., Wu, D., & Wang, Z. (2018, 8-13 July 2018). *Convolutional Neural Networks for Energy Time Series Forecasting.* Paper presented at the 2018 International Joint Conference on Neural Networks (IJCNN).
Krishnan, M., Jung, Y. M., & Yun, S. (2020, 11-13 March 2020). *Prediction of Energy Demand in Smart Grid using Hybrid Approach.* Paper presented at the 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC).
Kuo, P. H., & Huang, C. J. (2018). A high precision artificial neural networks model for short-Term energy load forecasting. *Energies, 11*(1). doi:10.3390/en11010213
LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278-2324. doi:10.1109/5.726791

LeCun, Y., Jackel, L. D., Boser, B., Denker, J. S., Graf, H. P., Guyon, I., . . . Hubbard, W. (1989). Handwritten digit recognition: applications of neural network chips and automatic learning. *IEEE Communications Magazine, 27*(11), 41-46. doi:10.1109/35.41400

Ma, Z., Christensen, K., & Jorgensen, B. N. (2021). Business ecosystem architecture development: a case study of Electric Vehicle home charging *Energy Informatics*.

Ma, Z., & Jørgensen, B. N. (2018, 19 - 22 Feb 2018). *Energy Flexibility of The Commercial Greenhouse Growers, The Potential and Benefits of Participating in The Electricity Market.* Paper presented at the IEEE PES Innovative Smart Grid Technologies North America (ISGT North America 2018), Washington, DC, USA.

Ma, Z., Sommer, S., & Jørgensen, B. N. (2016, 12-14 Oct. 2016). *The smart grid impact on the Danish DSOs' business model.* Paper presented at the 2016 IEEE Electrical Power and Energy Conference (EPEC), Ottawa, Canada.

Ma, Z., Værbak, M., Rasmussen, R. K., & Jøergensen, B. N. (2019, 22-25 July 2019). *Distributed Energy Resource Adoption for Campus Microgrid.* Paper presented at the 2019 IEEE 17th International Conference on Industrial Informatics (INDIN).

Nejat, P., Jomehzadeh, F., Taheri, M. M., Gohari, M., & Abd. Majid, M. Z. (2015). A global review of energy consumption, CO2 emissions and policy in the residential sector (with an overview of the top ten CO2 emitting countries). *Renewable and Sustainable Energy Reviews, 43*, 843-862. doi:https://doi.org/10.1016/j.rser.2014.11.066

Panapongpakorn, T., & Banjerdpongchai, D. (2019). *Short-Term Load Forecast for Energy Management Systems Using Time Series Analysis and Neural Network Method with Average True Range.* Paper presented at the 2019 1st International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics, ICA-SYMP 2019.

Pramono, S. H., Rohmatillah, M., Maulana, E., Hasanah, R. N., & Hario, F. (2019). Deep learning-based short-term load forecasting for supporting demand response program in hybrid energy system. *Energies, 12*(17). doi:10.3390/en12173359

Qi, X., Zheng, X., & Chen, Q. (2020). *A short term load forecasting of integrated energy system based on CNN-LSTM.* Paper presented at the E3S Web of Conferences.

Rahman, S., Alam, M. G. R., & Rahman, M. M. (2019, 18-20 Dec. 2019). *Deep Learning based Ensemble Method for Household Energy Demand Forecasting of Smart Home.* Paper presented at the 2019 22nd International Conference on Computer and Information Technology (ICCIT).

Ramokone, A., Popoola, O., & Awelewa, A. (2020, 25-28 Aug. 2020). *Application of occupancy-interlinked inhabitant behavior variables for improved energy and load profiles modeling.* Paper presented at the 2020 IEEE PES/IAS PowerAfrica.

Rodrigues, F., Cardeira, C., Calado, J. M. F., & Melício, R. (2017). Family houses energy consumption forecast tools for smart grid management. In *Lecture Notes in Electrical Engineering* (Vol. 402, pp. 691-699).

Rosato, A., Araneo, R., Andreotti, A., & Panella, M. (2019, 11-14 June 2019). *2-D Convolutional Deep Neural Network for Multivariate Energy Time Series Prediction.* Paper presented at the 2019 IEEE International Conference on Environment and Electrical Engineering and 2019 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe).

Rosato, A., Succetti, F., Araneo, R., Andreotti, A., Mitolo, M., & Panella, M. (2020, 29 June-28 July 2020). *A Combined Deep Learning Approach for Time Series Prediction in Energy Environments.* Paper presented at the 2020 IEEE/IAS 56th Industrial and Commercial Power Systems Technical Conference (I&CPS).

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323*(6088), 533-536. doi:10.1038/323533a0

Selvi, M. V., & Mishra, S. (2018, 13-15 Dec. 2018). *Investigation of Weather Influence in Day-Ahead Hourly Electric Load Power Forecasting with New Architecture Realized in Multivariate Linear Regression & Artificial Neural Network Techniques.* Paper presented at the 2018 8th IEEE India International Conference on Power Electronics (IICPE).

Selvi, M. V., & Mishra, S. (2020). Investigation of Performance of Electric Load Power Forecasting in Multiple Time Horizons With New Architecture Realized in Multivariate Linear Regression and Feed-Forward Neural Network Techniques. *IEEE Transactions on Industry Applications, 56*(5), 5603-5612. doi:10.1109/TIA.2020.3009313

Shikulskaya, O., Urumbaeva, O., & Shalaev, T. (2020). *Concept of Intelligent Energy Grid Control Based on Upgraded Neural Network.* Paper presented at the Proceedings - ICOECS 2020: 2020 International Conference on Electrotechnical Complexes and Systems.

Skatteministeriet. (2020). *Aftale mellem regeringen, Radikale Venstre, Socialistisk Folkeparti og Enhedslistenom: Grøn omstilling af vejtransporten*. Retrieved from https://www.skm.dk/media/7753/aftaletekst-aftale-om-groen-omstilling-af-vejtransporten.pdf

Saatwong, P., & Suwankawin, S. (2016). *Short-term electricity load forecasting for Building Energy Management System.* Paper presented at the 2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, ECTI-CON 2016.

Timur, O., Zor, K., Çelik, Ö., Teke, A., & İbrikçi, T. (2020). Application of statistical and artificial intelligence techniques for medium-term electrical energy forecasting: A case study for a regional hospital. *Journal of Sustainable Development of Energy, Water and Environment Systems, 8*(3), 520-536. doi:10.13044/j.sdewes.d7.0306

Torabi, M., & Hashemi, S. (2012, 2-3 May 2012). *A data mining paradigm to forecast weather sensitive short-term energy consumption.* Paper presented at the The 16th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP 2012).

Ves, A. V., Ghitescu, N., Pop, C., Antal, M., Cioara, T., Anghel, I., & Salomie, I. (2019, 5-7 Sept. 2019). *A Stacking Multi-Learning Ensemble Model for Predicting Near Real Time Energy Consumption Demand of Residential Buildings.* Paper presented at the 2019 IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP).

Vinagre, E., Gomes, L., & Vale, Z. (2015). *Electrical energy consumption forecast using external facility data.* Paper presented at the Proceedings - 2015 IEEE Symposium Series on Computational Intelligence, SSCI 2015.

Vinagre, E., Paz, J. F. D., Pinto, T., Vale, Z., Corchado, J. M., & Garcia, O. (2016, 6-9 Dec. 2016). *Intelligent energy forecasting based on the correlation between solar radiation and consumption patterns.* Paper presented at the 2016 IEEE Symposium Series on Computational Intelligence (SSCI).

Wollsen, M. G., & Jørgensen, B. N. (2015). Improved Local Weather Forecasts Using Artificial Neural Networks. In (Vol. 373, pp. 75-86). Cham: Springer International Publishing.

Yuce, B., Mourshed, M., & Rezgui, Y. (2017). A smart forecasting approach to district energy management. *Energies, 10*(8). doi:10.3390/en10081073

8. APPENDIX

8.1. Appendix B

```
# Import geosphere library
library("geosphere")
# Read Weather Data to get range of dates
df = read.csv('data/weather_features.csv')
dates = substr(df$Datetime, 1, 10)
# For loop to collect daylength at latitude 55.56
daylengths = list()
```

```r
for (date in dates){
tmp <- daylength(55.56, date)
daylengths[[date]] = tmp
}
# Write day lengths to csv
write.csv(daylengths, "data/daylengths.csv", row.names = TRUE)
```

8.2. Appendix C.1

Offset Split Method Function

```python
def ts_offset_split(dataframe, steps, lookback, horizon, batch_size, scaler='standard'):
    '''
    This pipeline function returns 3 Keras Timeseries Dataset Objects: train, validation, and test.
    The function first splits the data with the offset split method every 8th day.
    Afterward, the data is scaled by either the StandardScaler or MinMaxScaler from SK-learn library.
    Finally, the dataframe is split using the lookback and horizon parameters.
    '''
    # Offset 8th Day Split
    start = 0
    end = 168
    offset = 24
    training = []
    validation = []

    # For loop to extract every eight day
    for i in range(int((365+366)/8)): # 2 entire years (+1 leap year day)

        train = dataframe.iloc[start:end]
        val = dataframe.iloc[end:end+offset]
        training.append(train)
        validation.append(val)

        start += 192
        end += 192

    # Decide Splits for sets
    train = pd.concat(training)

    val = pd.concat(validation)

    train = train.append(dataframe[(dataframe.index.date > val.index.max()) & (dataframe.index.date < 
dt.date(2021,1,1))])

    test = dataframe[dataframe.index.date >= dt.date(2021,1,1)]

    tmpdf = pd.concat([train,val,test])
```

```python
# Scaler
if scaler == 'standard':
    X_scaler = StandardScaler()
    y_scaler = StandardScaler()
elif scaler == 'minmax':
    X_scaler = MinMaxScaler()
    y_scaler = MinMaxScaler()
elif scaler == None:
    print("Data has not been scaled.")
else:
    print('Please specify scaler: standard, minmax, or None')

# Training Split
start = lookback + horizon
end = start + train.shape[0]

X_train = train.values
y_train = tmpdf.iloc[start:end][['Value']]

if scaler != None:
    X_train = X_scaler.fit_transform(X_train)
    y_train = y_scaler.fit_transform(y_train)

# Validation Split
x_end = len(val) - lookback - horizon
y_val_start = train.shape[0] + lookback + horizon

X_val = val.iloc[:x_end]
y_val = tmpdf.iloc[y_val_start:y_val_start+x_end][['Value']]

if scaler != None:
    X_val = X_scaler.transform(X_val)
    y_val = y_scaler.transform(y_val)


# Test Split
x_end = len(test) - lookback - horizon
y_test_start = (train.shape[0] + val.shape[0]) + lookback + horizon

X_test = test.iloc[:x_end]
y_test = tmpdf.iloc[y_test_start:y_test_start+x_end][['Value']]

if scaler != None:
    X_test = X_scaler.transform(X_test)
    y_test = y_scaler.transform(y_test)
```

```
    # Batch Sequence Generators
    sequence_length = int(lookback/steps)

    dataset_train = timeseries_dataset_from_array(
        X_train, y_train,
        sequence_length=sequence_length,
        sampling_rate=steps,
        batch_size=batch_size,
        shuffle=True
    )

    dataset_val = timeseries_dataset_from_array(
        X_val, y_val,
        sequence_length=sequence_length,
        sampling_rate=steps,
        batch_size=batch_size,
        shuffle=True
    )

    dataset_test = timeseries_dataset_from_array(
        X_test, y_test,
        sequence_length=sequence_length,
        sampling_rate=steps,
        batch_size=batch_size,
        shuffle=False
    )

    return dataset_train, dataset_val, dataset_test
```

8.3. Appendix C.2

Functions for the Final Model Training Pipeline

```
# Function to print metrics
def metrics(true, pred):
    print(f'MSE: {np.mean(mean_squared_error(true, pred))}')
    print(f'RMSE: {np.sqrt(np.mean(mean_squared_error(true, pred)))}')
    print(f'MAE: {np.mean(mean_absolute_error(true, pred))}')
    print(f'MAPE: {np.mean(mean_absolute_percentage_error(true, pred))}')

# function to return error metrics
def get_metrics(true, pred):
    mse = np.mean(mean_squared_error(true, pred))
    rmse = np.sqrt(np.mean(mean_squared_error(true, pred)))
    mae = np.mean(mean_absolute_error(true, pred))
    mape = np.mean(mean_absolute_percentage_error(true, pred))
    return mse, rmse, mae, mape

# function to load data in the training pipeline
def load_data(config, splitmethod):
    # Load csv & parse dates to datetime index
    data = pd.read_csv(f'../data/processed/{config.dataset}_features.csv',
index_col='Datetime', parse_dates=['Datetime'])
```

```
    # Select Features
    data = data[['Value', 'sunshine_mins', 'airtemp_c', 'daylength_hrs', 'wkdy_sin',
'wkdy_cos', 'wknd', 'mnth_sin', 'mnth_cos']]

    if splitmethod == 'offset':
        train, val, test = processing.ts_offset_split(data,
                                steps=steps, lookback=config.lookback,
                                horizon=horizon, batch_size=config.batch_size,
                                scaler='standard')

    elif splitmethod == 'standard':
        train, val, test = processing.create_datasets(data, split=config.splitrate,
                                steps=steps, lookback=config.lookback,
                                horizon=horizon, batch_size=config.batch_size,
                                scaler='standard')

    return train, val, test

# function to build model in training pipeline
# all parameters are collected from the wandb-configuration dictionary
def build_model(config):
    # Builds the model for training
    model = Sequential(name='Hybrid-CNNRNN-Kolding')

    model.add(Input(shape=(config.lookback, config.num_features), name='InputLayer'))

    # CNN Block
    model.add(Conv1D(filters=config.cnn_layer_size_1,
        kernel_size=config.kernelsize,
        activation=config.activation_cnn,
        name='ConvolutionalLayer'))
    model.add(MaxPooling1D(pool_size=2, name='MaxPoolingLayer'))

    # RNN Block
    model.add(GRU(config.gru_layer_size_1,
        return_sequences=False,
        activation=config.activation_gru,
        name='GRURecurrentLayer'))
    model.add(Dropout(config.dropout, name='Dropout'))

    model.add(Dense(1, name='OutputLayer'))

    opt = config.optimizer
    if opt == 'sgd':
        optimizer = SGD(learning_rate=config.learning_rate, momentum=config.momentum)
    model.compile(optimizer=optimizer, loss='mse', metrics=['mae'])
    return model


# function to apply parameters to the wandb-config
def log_defaults():
    # This functions applies the parameters to the configuration
    # The following values resulted in the strongest model
    wandb_config = {
        'num_features' : 9,
        'epochs' : 100,
        'batch_size' : 224,
        'optimizer' : 'sgd',
        'dropout' : 0.2,
        'lookback' : 744,
        'activation_cnn' : 'relu',
        'activation_gru' : 'tanh',
        'kernelsize': 3,
        'cnn_layer_size_1' : 56,
```

```
    'gru_layer_size_1' : 96,
    'learning_rate' : 0.04,
    'momentum' : 0.9,
    'dataset' : 'kolding',
    'splitmethod': 'offset',
}
return wandb_config
```

8.4. Appendix C.3

Training Pipeline

```
# Additional Parameters
steps = 1 # timesteps: 1 hour
horizon = 1 # the target hour in the future we want to predict 1 hour ahead

# Initialize wandb and the wandb-config dictionary
wandb.init(config=log_defaults(), group='cnnrnn-final-train-kolding', project='thesis')

# Build model using the config
model = build_model(config=wandb.config)

# Load data and setup callback functions
train, val, test = load_data(config=wandb.config, splitmethod=wandb.config.splitmethod)

reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.6, patience=5)
ckpt_path = "../model/models/cnngru_final-kolding.h5"
estp = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)

ckpt = ModelCheckpoint(
    monitor='val_loss',
    filepath=ckpt_path,
    verbose=1,
    save_weights_only=True,
    save_best_only=True
)

callbacks = [WandbCallback(), reduce_lr, ckpt, estp]

# Train the model
history = model.fit(
    train,
    epochs=wandb.config.epochs,
    validation_data=val,
    callbacks=callbacks
)

# Evaluate the model
eval = model.evaluate(test)
wandb.log({'test_loss': eval[0]})
wandb.log({'test_mae': eval[1]})
print(f'\nTest Loss: {eval[0]} - Test MAE: {eval[1]}')
```

8.5. Appendix C.4

Creating the Predictions from the Final Model

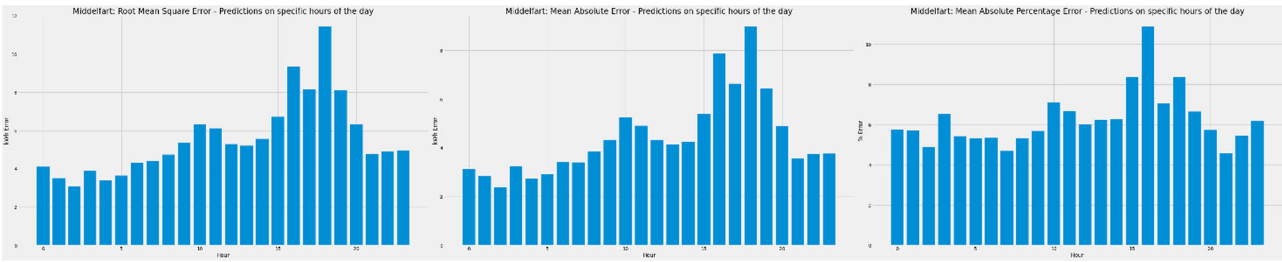
```
# predict and inverse scale
pred = model.predict(test)
inverse_scaled_pred = (pred * train_std) + train_mean
inverse_scaled_pred
# Get true values from test set
y_true = kld_test.iloc[744+1:len(kld_test) - 744 + 1][['Value']]
# Create DataFrame of True and Pred for easier plotting
pred_df = y_true.copy()
```

```
pred_df.rename(columns={'Value':'True'}, inplace=True)
pred_df['Pred'] = inverse_scaled_pred
```

8.6. Appendix D

Kolding Day & Time-Based Errors

Middelfart Day & Time-Based Errors




Middelfart: Root Mean Square Error - Predictions on specific weekdays
Weekdays
Middelfart: Mean Absolute Error - Predictions on specific weekdays
Weekdays
Middelfart: Mean Absolute Percentage Error - Predictions on specific weekdays
Weekdays
Middelfart: Root Mean Square Error - Predictions on specific weeks
Week Number
Middelfart: Mean Absolute Error - Predictions on specific weeks
Week Number
Middelfart: Mean Absolute Percentage Error - Predictions on specific weeks
Week Number
Middelfart: Root Mean Square Error - Predictions on specific months
Month
Middelfart: Mean Absolute Percentage Error - Predictions on specific months
Month
Middelfart: Mean Absolute Error - Predictions on specific months
Month