2022

Load Forecasting and Synthetic Data Generation for Smart Home Energy Management System

Mina Razghandi
University of Central Florida

LOAD FORECASTING AND SYNTHETIC DATA GENERATION FOR SMART HOME ENERGY MANAGEMENT SYSTEM

by

MINA RAZGHANDI
M.S. AmirKabir University of Technology, 2017
B.S. University of Tehran, 2014

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term
2022

Major Professor: Damla Turgut

ABSTRACT

A number of recent trends, such as the increased power consumption in developed and developing countries, the dangers associated with greenhouse gases, the potential shortages of fossil fuels, and the increasing availability of solar and wind energy act as motivating factors for the development of more intelligent and efficient systems both on the power provider as well as the consumer side.

One of the most important prerequisites for making efficient energy management decisions is the ability to predict energy production and consumption patterns. While long-term forecasting of average consumption had been extensively used to direct investments in the energy grid, short-term predictions of energy consumption became practical only recently. Most of the existing work in this domain operates at the level of individual households.

However, the availability of historical power consumption data can be an issue due to concerns such as privacy, data size or data quality. Researchers have been provided with synthetic smart home energy management systems that mimic the statistical and functional properties of the actual smart grid in order to improve their access to public system models. Through developing time series to represent different operating conditions of these synthetic systems, the potential of artificial smart home energy management system applications will be further enhanced.

The work described in this dissertation extends the ability to predict and control power consumption to the level of individual devices in the home. This work is made possible by several recent developments. Internet of things technologies that connect individual devices to the internet allows the remote tracking of energy consumption and the remote control and scheduling of the devices. At the same time, progress in artificial intelligence and machine learning techniques improve the accuracy of predictions. These components often form the basis of smart home energy management systems (HEMS).

One of our insights that facilitates the prediction of the energy consumption of individual devices is that the history of consumption contains important information about future consumption. Thus, we propose to use a long short-term memory (LSTM) recurrent neural network for prediction. In a second contribution, we extend this model into a sequence-to-sequence model which uses several interconnected LSTM cells on both the input and the output sides. We show that these approaches produce better predictions compared to memoryless machine learning techniques.

The prediction of energy consumption delivers maximum value when it is integrated with the active component of a HEMS. We design a reinforcement learning-based technique where a Q-learning model is trained offline based on the prediction results. This system is then validated only using real data from PV power generation and load consumption.

Considering the scarcity of data among the smart grid users, in our third contribution, we propose the Variational Autoencoder Generative Adversarial Network (VAE-GAN) as a smart grid data generative model capable of learning various types of data distributions, such as electrical load consumption, PV power production and electric vehicles charging load consumption, and generating plausible sample data from the same distribution without first performing any pre-training analysis on the data. Our extensive experiments have shown the accuracy of our approach in synthesizing smart home datasets. There is a high degree of resemblance between the distribution of VAE-GAN synthetic data and the distribution of real data. The next step will be to incorporate Q-learning for offline optimization of HEMS using synthetic data and to test its performance with real test data.

ACKNOWLEDGMENTS

There are many individuals who have had a profound impact on my life and educational career. I could never possibly list all the names of those who have inspired me or touched my life over the years. Nevertheless, I wish to acknowledge the people who had a significant impact on my doctoral experience.

To begin with, I would like to offer a special thanks to my dissertation committee, Professor Changchun Zou, Professor Qun Zhou Sun, and Professor Kelly Stevens, for the guidance and support they provided during my doctoral pursuits. As part of my gratitude, I would like to thank my dedicated and compassionate dissertation chair, Professor Damla Turgut, who has been very supportive and encouraging throughout my dissertation, for believing in me and my work, and for her tremendous support and assistance. Moreover, I wish to express my appreciation to my collaborators at the University of Ottawa, Dr. Hao Zhou, and Professor Melike Erol-Kantarci. Their keen eyes and advice helped me to be a better writer and scholar.

During this journey, my family and friends never stopped supporting me, encouraging me, and being there for me even when times were tough. On this note, I would like to express my deepest gratitude to them. It was through the patience and love of my husband, Pooya, that I was able to endure my worst times and found a new definition of support for me and for taking care of our fur baby while giving me space and time to research. I could not imagine standing where I am today without him by my side. I owe my education to my parents, Zahra and Esmaeel, who taught me the value of education and taught me that success requires hard work, and have given me the tools to grow and shine.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

The term *smart home* refers to a residence where appliances and services are connected through a communication network, allowing control, monitoring, and remote access [24]. The need for sustainable energy solutions in the form of renewable energy sources and energy-efficient technologies increases the need for considering renewable energy sources in smart home design. Recent advances in smart grid research include new technologies and strategies for managing energy generation, storage, supply, and demand [3]. Smart homes, as a critical component of the smart grid, are projected to increase household energy efficiency, save energy costs, and improve user comfort. Utilities and smart home controllers will be able to glean information from smart meters for use in forecasting consumption and generation, demand-side management, and economical power dispatch [58]. Thus, acquiring fine-grained data about the living environment is becoming an essential precondition for a smart home.

Managing the electric load is an important task of a smart home. A residential energy load can be classified as deferrable or non-deferrable. A deferrable load is one whose demand can be managed over time while a non-deferred load cannot be controlled in its timing [77]. A smart home energy management system (HEMS) monitors, controls, and manages home devices, including solar panels, smart appliances, and energy storage systems. For example, the controller might reduce the electricity costs by shifting some deferrable loads from the peak energy price period to the off-peak period by controlling the on/off status of smart appliances such as the washing machine and the dishwasher [71].

HEMS differs from smart meters in that it provides users with direct and immediate insights into their energy consumption. By integrating smart meters with HEMS, home appliances can be monitored and controlled in real-time via the in-home display. More academic and industrial researchers

are exploring the integration of smart meters and HEMS to save energy. HEMS are being evaluated in order to determine if they can reduce energy consumption by encouraging behavior change, and if so, whether their implementation and maintenance costs are justified by the energy they can save. [64] refers to effectiveness as positive net energy savings where $e_{saved} > e_{invested}$ whereas other research considers effectiveness as energy reduction in a short period of time. The authors consider three different HEMS:

- Energy Monitor: Has a small monitor that provides real-time feedback on overall energy consumption.
- Multifunctional HEMS: Provides historical and real-time configuration feedback about overall gas and electrical consumption.
- Energy Management Device: The energy management device gives real-time and historical feedback on the electricity consumption of individual appliances.

[64] states points that need to be taken into consideration. HEMS has a technical lifespan and has environmental costs during the lifespan, however, users might decide to discontinue using HEMS for saving energy. Moreover, HEMS is reliant on the availability of peripheral devices outside the system boundary such as smart meters, routers, etc. There is no doubt that all these HEMS have the potential to achieve positive net energy savings in their technical lifespan and to increase ROI over the long run, but the chances are strongly dependent on variables into which there is currently too little insight [64].

In another research, Nilsson et al. [45] investigate the effects of feedback and smart home features on energy consumption, household perceptions, use, and actions related to energy consumption, and to discuss the perception of barriers to energy consumption behavior change. [45] adopt a consumer-oriented interdisciplinary approach by combining quantitative and qualitative methods.

They interview 14 households and analyse their energy consumption measurements. The findings suggest that energy savings are highly reliant on the resident's lifestyle, willingness, and the ability to engage with the information and features provided and can vary from 49.0% in saving energy to a 42.9% increase in energy consumption. According to the study, high-income, highly educated households were more likely to reduce their energy consumption to benefit the environment. The study participants stated that HEMS helped them better understand what is considered standard energy consumption, heightened their awareness of unnecessary energy use, and made them feel more comfortable in their own homes. But there were many obstacles to behavioral change, including a lack of knowledge, a sense of lack of control, as well as attitudes and values. In other words, HEMS failed to achieve its goal of increasing knowledge about energy consumption or providing solutions to save energy. This left residents with the impression they could not do anything about it. Furthermore, residents were justified in believing that certain types of consumption that are related to comfort and well-being are acceptable [45].

A number of studies have demonstrated the importance of combining energy consumption data from HEMS with information about user behaviors and attitudes so that energy efficiency behaviors can be assessed holistically. A study was conducted to investigate how household energy consumption was affected, how daily consumption patterns changed, and how peak periods were reduced in homes that installed HEMS, by comparing real-life consumption values from 10 different households in winter months to the same months the following year [62]. Researchers found that households could save up to 30% if they were willing to compromise their comfort and be responsible for their energy consumption. If not, there is still the possibility of obtaining some level of savings. However, the level of energy savings within an individual household can vary from month to month [62].

However, the HEMS is highly affected by uncertainties. For energy generation, the output power of solar panels changes with time and weather. On the other hand, consumers' living habits differ

from one another: smart home residents have preferences for how often and how much energy they consume depending on external factors such as temperature, humidity or holidays [85, 11]. Short-term forecasting of household loads is closely tied to understanding consumer habits. Compared with load forecasting for a community or a commercial building, it is more challenging to forecast the short-term load for a household [55, 27, 6, 4, 84] due to the stochastic habits of consumers and the varying status and types of household appliances.

Understanding and predicting the energy consumption patterns in a household can provide benefits to all stakeholders [67]. Accurate and robust household load forecasting benefits both the utility company and individual consumers. The utility company, which might be using a smart grid, could better manage the electricity distribution and ancillary services while offering dynamic pricing to reduce peak demand. For the individual consumers, predicting the load allows the identification of energy loads that can be shifted to off-peak hours, thereby reducing the energy bill.

Based on the prediction period, load forecasting is classified as:

1. *Very Short-Term Load Forecast (vSTLF)*: forecasting the load for the next several minutes.
2. *Short-Term Load Forecast (STLF)*: predicting the load from the next several hours to a week ahead.
3. *Medium-Term Load Forecast (MTLF)*: predicting the load from a week to a year ahead.
4. *Long-Term Load Forecast (LTLF)*: predicting the load over the timespan of several years.

One of the techniques proposed in the literature for reducing uncertainty is based on using classification to group similar load curves [55]. Another technique uses a pre-processing transformation, e.g., Fourier transforms or wavelet analysis, aiming to extract the regular consumption pattern from the overall electricity load [74]. It is worth noting that, as new household smart appliances

with varied energy consumption regimes are adopted by consumers, the residential electrical load profile deviates from traditional consumption profiles [27].

In recent years, household load forecasting took advantage of several modern AI techniques, such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), and deep belief networks (DBNs) [6].

Neural networks are capable of modeling the nonlinear and non-stationary nature of energy consumption. For instance, CNN has been used for short-term load forecasting in [53, 4], and the results show that CNNs outperform support vector machines (SVM). Long short term memory (LSTM) networks had been applied for load forecasting in [28, 74, 84], and were found to have a lower prediction error compared with traditional methods.

Deep learning models for aggregated load demand forecasting have received much attention during the last years [32, 33]. However, forecasting energy demand for appliances in a household whose use is largely determined by user preferences, such as washing machines, dishwashers, and other appliances, remains a challenge.

Most previous works focused on forecasting the load at the level of the whole household. Appliance-level load forecasting for residential consumers is of significant importance for the modern power system, particularly considering the dynamic loads such as electric vehicles that will stress the distribution system and consequently call for fine-grained control of micro-loads by home energy management systems. Therefore, mid-term and short-term load forecasting is essential for distribution system operation and control, while long-term prediction plays a key role in infrastructure planning [28].

Our study starts with the insight that while the total load is important, individual energy-saving or load-shifting actions taken by the users, such as postponing the operation of the dishwasher to the

night hours, will usually affect individual appliances. Thus, our study focuses on the appliance-level short-term load forecasting model for residential homes, which has a higher level of uncertainty. In Chapter 3, we take advantage of recent developments in deep learning techniques to create an LSTM recurrent neural network-based predictor. We train the network based on historical data on residential home appliances' energy usage and estimate energy usage for a given appliance in the short term. This problem is more complicated than simply training a number of LSTM-based models for each household appliance. We found that training a single model for all appliances together is more robust and scalable.

In Chapter 4, our appliance-level short-term load forecasting model is further developed into an LSTM-based sequence to sequence (Seq2Seq) learning model. This technique was initially proposed in the context of machine translation in [59], where a multilayered LSTM network maps a sequence of input to a fixed-size vector and then, another multilayered LSTM network maps the fixed-sized vector to the sequence of target words. In this study, we use an LSTM network to map the sequence of past 24-hour energy consumption values to a fixed-size vector, then detect the appliance type, regenerate the input sequence in reverse form from the fixed-sized vector using another LSTM network, and produce a sequence of energy consumption values for the next hour from the fixed-length using another LSTM network. Seq2Seq learning has been shown to effectively deal with variable input and output sequence lengths and to be more effective at feature extraction. By visualizing the predictions of our network and observing prediction error, we are able to see that our proposed model can distinguish the appliance type by its trend and learns appliances' typical usage duration.

Another aspect of our work focuses on finding optimal policies for the HEMS control policy. Given the uncertainty of energy generation and consumption, it is hard for traditional methods to build a dedicated optimization model. We propose a model-free reinforcement learning (RL) technique where this complexity is avoided by using a uniform Markov decision process (MDP).

The agent interacts with the environment and gets rewards by choosing different actions. By exploring possible action combinations, the agent learns the best action sequence to maximize the long-term reward.

In traditional approaches for comparing prediction techniques, the comparison is done through metrics such as Root Mean Squared Error (RMSE), without considering the consequence of prediction errors on the control systems. In this study, we also investigate the performance of control operations in relation to various prediction techniques. First, we compare several forecasting algorithms, including Vector Auto-Regressive Moving Average (VARMA), Support Vector Regression (SVR), LSTM, and Seq2Seq. Q-learning is then applied for HEMS, which utilizes the forecasting results for offline training. Finally, we test the trained Q-learning with real-world data and observe how different prediction methods affect the operation cost.

It has become an increasingly common practice for artificial intelligence and statistical analysis for demand response and energy management tasks that require accurate short-term (second to minute scale) representations of load behavior. While data availability for transmission systems has generally been straightforward, the issues related to data collection, security, and privacy have presented a challenge to their widespread dissemination and accessibility. Smart home / Smart Grid data is subject to confidentiality, integrity, availability, authenticity, and authorization security concerns. Consequently, there are relatively few high-quality datasets that are available publicly due to data availability, scale, scope, and size limitations of the data, as well as concerns over privacy and security [82, 26].

Furthermore, the size and quality of real-world data can also present a bottleneck when applying data science techniques to the smart grid [26]. This renders the generation of synthetic data to be an attractive alternative. Once the generated data is available, machine learning algorithms can help make decisions, for example, to determine the optimal time for implementing demand response,

charging an EV, etc [87].

The existing works around synthetic data generation for smart homes can be divided into two approaches: model-based and data-driven methods. Model-based generated datasets about smart homes have been developed statistically and mathematically, including Markov chain [13, 18], statistical model [30], and physical simulator-based methods [38]. A major drawback of these methods can be attributed to targeting a limited number of problems, applications tailored to specific scenarios, requiring extensive knowledge to build a dedicated generation model, and the lack of scalability. These methods fail to capture the behavior underlying the system.

Recent studies, on the other hand, have utilized deep learning-based approaches as a data-driven method that can be applied to large-scale datasets and can be conducted directly on raw data, without the need for further analysis. These methods avoid the complexity of building a dedicated physical operation model of household devices and consequently increase flexibility by eliminating tedious assumptions. Recently, machine learning techniques provided useful tools for synthetic data generation, with Generative Adversarial Networks (GAN) being one of the most promising approaches[19]. For instance, [83] proposed a deep GAN-based method to generate synthetic data for energy consumption and generation. The main idea behind the GAN network is to use a discriminator to indirectly train the generator network to produce synthetic data. The generator must deceive the discriminator in order to not distinguish between fake and real samples to reach an equilibrium point.

In Chapter 5, we propose a novel Variational Autoencoder GAN (VAE-GAN) technique for the synthetic time series data generation of smart homes. In contrast to the schemes mentioned above, this strategy enables the learning of various types of data distributions in a smart home, including electric load profiles, photovoltaic power (PV) generation, and electric vehicle (EV) charging load consumption, and can subsequently generate plausible samples from those same distributions

without carrying out any prior analysis before the training phase. Compared to the vanilla GAN architecture, the VAE-GAN architecture is favored in this model due to its resistance to mode collapse, the occurrence in which the generator learns a false output that deceives the discriminator and produces that output repeatedly and over and over again. As an additional advantage, VAE-GAN allows us to fine-tune the latent space that influences the generated output, as opposed to vanilla GAN in which the generator maps the input noise to the generated output. Furthermore, we compare one model-based and two data-driven generative models, including Gaussian Mixture Model (GMM), vanilla GAN, and VAE-GAN.

Using the synthetic data generated through this technique, we propose a Q-learning-based smart home energy management system (HEMS). First, we use the generated data for the off-line training of Q-learning HEMS agents, which aims to maximize the long-term management profit. Then, the trained agents are tested in a real-world data-based environment, which is considered an online operation test. Finally, we compare online profits to further investigate how the data generation method will affect the HEMS performance. The idea behind this scheme is that we assume good-quality synthetic data in the off-line training can better prepare the agent for real-world operations due to the data similarities[51]. Consequently, the defined Q-learning-based HEMS can be used to further demonstrate the synthetic data quality.

Compared with conventional model-based optimization algorithms such as convex optimization, the reinforcement learning (RL) based method can significantly reduce the complexity of defining a sophisticated optimization model since the optimization problem can be transformed into a unified Markov decision process (MDP) scheme. On the other hand, although there have been a huge amount of HEMS models, most existing works assume a perfect environment for data collection and algorithm training[61]. By contrast, the proposed Q-learning HEMS uses synthetic data for algorithm training, and it overcomes the data availability bottleneck for applying machine learning techniques to the smart grid field.

The remainder of this dissertation is organized as follows. Chapter 2 introduces the related work. Chapter 3 describes the appliance-level short-term load forecasting model based on the LSTM network. Chapter 4 improves the LSTM-based load forecasting model to a sequence to sequence learning and employs the forecasting results to off-line optimize Q-learning-based HEMS. Chapter 5 introduces the proposed smart home synthetic data generation model and Q-learning-based HEMS model. Chapter 6 concludes the dissertation.

CHAPTER 2: LITERATURE REVIEW

Our discussion in this chapter will cover some of the most recent studies conducted in the field of short-term load forecasting and energy management systems in smart homes. As part of our discussion, we will discuss different approaches and methods used in short-term load forecasting along with a few examples in each area to illustrate how our approach differs from the studies mentioned. Moreover, we will discuss recent developments in artificial intelligence in regard to smart home energy management systems as well.

2.1 Short-Term Load Forecasting

Deep learning has shown tremendous potential in many fields over the past few years, which will open up new possibilities for load forecasting as well. Most recently, modern AI techniques, especially machine learning and deep learning, became the mainstream techniques for load forecasting [7]. Most literature use one of the following methods of load forecasting:

1. **Statistical methods**: Autoregressive Integrated Moving Average (ARIMA), Gray Model (GM), the Kalman Filtering, etc.
2. **Machine learning methods**: Linear Regression, Support Vector Regression, and AutoRegressive Integrated Moving Average (ARIMA), etc.
3. **Deep learning algorithms**: Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) and etc.
4. **Probabilistic forecasting**: Quantile Regression, Density Regression, etc. Probabilistic forecasts may provide more comprehensive information on potential uncertainties rather than

predicting the expected load throughout time.

5. **Hybrid methods**: Combining CNNs with Gated Recurrent Units.

2.1.1 Statistical Methods

Most common statistical forecasting methods include Autoregressive Integrated Moving Average (ARIMA) [2, 44], Gray Model (GM) [23, 9], and the Kalman Filtering method [5]. In comparison to machine learning, deep learning, and hybrid approaches, these models lack the ability to take advantage of daily life routines and activities to provide more accurate predictions.

2.1.2 Machine Learning-based Load Forecasting

Wu et al. [72] present a gradient boosting-based multiple-kernel learning framework for the electrical load forecasting problem. The gradient boosting method allows predictions of regression with several regressors combining results. They tackle the issue of data scarcity by integrating transfer learning for training the model. The outcomes from mean absolute percentage error (MAPE) and standard error show that the proposed model has better performance in comparison with Linear Kernel, SVR, Gaussian Cycle (GP), and Long Short-Term Memory (LSTM).

Peng et al. [48] explore load predictability Small-and-Medium Enterprise compared to residential homes in different aggregation levels by approximate entropy measure (ApEn). The decreasing pattern of ApEn shows that Small-and-Medium Enterprise (SME) loads are much more predictable, even on an individual basis. At the same time, predictive accuracy improves for residential homes as aggregation levels increase. Furthermore, Normalized Mean Absolute Error (NMAE) results show that linear regression, multi-layer perceptron (MLP), and SVR have better performance compared to Gradient Boosted Regression Tree (GBRT) for SME loads and residential datasets

respectively. Only the degree of aggregation did not affect linear regression, SVR, and MLP performance for residential loads.

A sparse coding-based learning approach for household electricity demand forecasting has been proposed by Yu et al. [80], and 10% higher accuracy has been observed in comparison to the existing studies.

2.1.3 Deep Learning-based Load Forecasting

Tang et al. [60] forecast energy consumption by combining two ARIMA and two LSTM models with a fully connected layer as the last layer to estimate energy consumption based on the outcome of these submodels. After data preparation, four sub-models are trained, including ARIMA I and LSTM I for continuous days, and ARIMA II and LSTM II for related days. A fully-connected layer estimates the final values based on the outcome of these four sub-models. The authors used Toronto's daily load consumption data to evaluate their model. Comparing the MAPE, Worst Relative Error (WRE), and RMSE metrics for this model against other standard forecasting models, including MLP, SVR, ARIMA, and LSTM, shows the superior performance of the proposed model.

Li et al. [33] propose a Convolutional Long Short-Term Memory based neural network with selected autoregressive features to improve floor-level load prediction accuracy for residential buildings.

Kong et al. [28] offer a residential load forecasting framework incorporating the recurrent neural network (RNN) to overcome the volatile features of the household load consumption pattern. LSTM is the most common RNN algorithm. Since LSTM produces the best results for sequences with long-term dependencies, it would be an excellent choice to forecast a single household consumption pattern that is entirely different from the system-level load consumption (load aggrega-

tion over a grid). The authors evaluated the performance of this algorithm with the conventional neural backpropagation network (BPNN), k-nearest neighbors (KNN), and the hybrid forecasting framework (IS-HF) with the Mean Absolute Percent Error (MAPE) metric.

Kong et al. [29] propose a Deep Belief Network (DBN) to enhance the load forecasting accuracy considering demand-side management. Using energy prices, the authors first construct a price Henkel matrix in the data optimization process and use a Gray Relational Analysis (GRA) to pick the most correlated variables. At the model optimization phase, these data go into a prediction model combining a Boltzmann restricted Gauss-Bernoulli machine (GB-RBM) and a Boltzmann restricted Bernoulli-Bernoulli machine (BB-RBM). They then used a mixed pre-trained technique to fine-tune the network parameters. They then used a mixed pre-trained technique to fine-tune the network parameters and test their model output using MAPE and RMSE metrics with ARIMA, LSSVM, and simple DBN.

Wang et al. [65] proposed Sparse Continuous Conditional Random Fields (sCCRF) to learn customer behavior living in a smart grid through supervised learning that contributes to precise load forecasts. The authors use the k-means method to cluster similar customers with similar behavior patterns. Subsequently, they fine-tune an sCCRF model for each cluster of customers to learn the behaviors more precisely. The cumulative load for the smart grid is the summation of the load forecast for individuals. The authors have compared their model performance with ARIMA, Support Vector Regression (SVR), CCRF, CNN, and LSTM methods in terms of the MAPE metric.

Kong et al. [27] used deep learning methods to predict and disaggregate energy consumption for home Type II appliances without sub-meter information for the target appliances. Type II appliances have multiple states of power consumption, complex state transitions, and multiple operating modes such as washing machines and dishwashers. The authors use a VGG-16 pre-trained CNN as their base network structure with a knowledge base of appliance profiles to learn the patterns

for different brands and models of target appliances and train another CNN for post-processing on the results of the first network. F1-score is the performance metric used for assessing this model over testing it on the data of 5 different households. Non-intrusive load monitoring (NILM) is a technique used to predict operation time for circuit-connected devices to estimate overall energy consumption. Such technologies are more compliant with existing smart meters and cost less for the consumer because there is no need for individual meters for each unit. They also make it easier to provide insights into personalized service recommendations according to each consumer's behavior.

Welikala et al. [70] presented a None-intrusive load monitoring (NILM) method for aggregate load forecasting in residential homes using appliance usage patterns. They used active traces of individual appliances to extract usage patterns through an initial a priori unbiased NILM approach. A fuzzy framework is used to measure the on/off time probabilities. Such data is fed into a priori-biased NILM method to estimate the aggregated load in a residential home for the next 5 minutes. The authors used two standard load disaggregation datasets to test their system. The metrics included Appliance Combination Identification Accuracy (APCIA), F-Measure, Total Power Correctly Assigned (TPCA), and Average Execution Time (AET) to compare the model output with disaggregation algorithms such as Simple NILM, Hidden Markov Model-based algorithms, and Sparse Coding based techniques.

Deep learning has been used in non-intrusive load monitoring where no sub-metered information is needed to estimate the demand of individual appliances [27, 36, 81]. Shomski et al. [56] propose a sequence-to-sequence learning method to predict the total energy consumption of commercial buildings with 1 hour and 15 minutes resolution.

2.1.4 Probabilistic Load Forecasting

Wang et al. [66] have combined Quantile Regression Neural Network (QRNN), Quantile Regression Random Forests (QRRF), and Quantile Regression Gradient Boosting (QRGB) methods and choose the process with the most optimized weights. More precisely, the ensemble problem is redefined as a Linear Programming (LP) optimization problem after training each model and calculating the most optimized weights. The goal of this optimization problem is to minimize pinball loss with optimal weights for the individual probabilistic forecasts in the solution. The results show that the proposed ensemble approach effectively reduces the pinball loss forecasting efficiency as opposed to models including Best Individual (BI), Naiive Sorting (NS), and Median Value (MED).

Feng et al. [17] introduced STLF-QMS, a mixture of deterministic and probabilistic load forecasting models with Q-learning dynamic model selection (QMS). The STLF-QMS framework has a DLF model pool, which includes Artificial neural network (ANN) methods, SVR methods, gradient boosting machine (GBM) methods, and Random Forest (RF). It also has a PLF model pool, including normal, Gamma, Laplace, and noncentral-t distributions DLF cumulative distribution function (CDM)-derived quantiles at each time step. QMS first selects the DLF model with the best output at each time step and determines the most effective PLF model. The authors used normalized mean absolute error (nMAE), mean absolute percentage error (MAPE) and normalized root mean square error (nRMSE) to test the efficiency of DLF models and PI coverage probability (PICP) and interval score (IS) for PLF models.

Yang et al. [75] applied Bayesian deep learning to implement probabilistic load forecasting, and a clustering-based pooling method has been designed to prevent overfitting to improve the predictive performance. The overfitting problem has been also considered by Shi et al. [55], where a novel pooling deep RNN is proposed to directly learn the uncertainty of load forecasting.

2.1.5 Hybrid methods

There are also a few hybrid methods proposed. For instance, LSTM has been combined with the stationary wavelet transform technique for individual household load forecasting in [74]. CNN and Gated Recurrent Units have been unified in [53], resulting in lower computational complexity and higher prediction accuracy.

Haq et al. [21] classify electrical appliance load into daily, weekly, monthly, and total energy consumption and use a hybrid machine learning method for load forecasting and peak demand.

2.2 Smart Home Energy Management System (HEMS)

Energy management systems for smart homes are automated systems that are able to manage the relationship between energy demand and available energy resources, thereby reducing energy costs without compromising user comfort. In these systems, some parameters such as tariffs, load forecasting, and level of importance are used to formulate optimization problems that can then be used to calculate the cost under certain conditions under certain factors. This optimization problem can be solved using various mathematical techniques and deep learning techniques. Data collecting and monitoring, data processing, load forecasting, optimization, and execution cycles are often included in HEMS frameworks. [8].

Lu et al. [39] propose an hour-ahead demand response method for HEMS to minimize the energy cost, which involves a neural network for energy price prediction.

An IoT-based self-learning HEMS, proposed by Li et al. [34], utilizes a long-short-term-memory network for energy price prediction, price clustering, and power alert systems.

A multi-agent RL-based method for HEMS is introduced by Xu et al. [73] to reduce energy cost,

in which the feed-forward neural network will predict energy price and solar generation.

Yousefi et al. [79] propose an energy management system for residential buildings based on plug-in EVs and PV power as energy sources, combined with a heat pump and load demand. The optimization problem in their HEMS system is solved by stochastic model predictive control (MPC). Due to its ability to operate in real-time, MPC has preferred over load forecasting.

Yao et al. [77] employ mixed-integer linear programming (MILP) as a means of scheduling load consumption and managing power dispatch from the utility grid under one single optimization problem. Smart homes are assumed to have a utility grid with dynamic electricity prices and interruptible, uninterruptible, and time-varying load characteristics. The MILP technique has been used in a number of other studies, including [42, 40, 20].

Lokeshgupta et al. [37] applied game theory techniques to optimize the performance of a smart home energy management system. A player's objective is to reduce energy costs as well as reduce the peak load demand of the home.

Mathew et al. [41] proposed a Deep Reinforcement Learning model with prioritized experience sampling (PQDN-DR) in which the problem of load shifting is simulated as a game. Also to better converge the DRL model to near-optimal strategies a new reward system is implemented. Moreover, the agent is guided in the exploration phase with a DR-adapted Epsilon Greedy Policy for faster convergence.

Dimitroulis et al. [14] employes fuzzy logic to optimize an intelligent EMS in which a variety of power components such as solar and wind generators, battery energy system (BES), electric vehicle (EV) load, dynamic electricity pricing, and tariffs are integrated.

Yang et al. [76] propose a home energy management strategy capable of managing the power supply of a residential house during a planned outage period. With this strategy, the residential

photovoltaic solar panels (PVs), coupled with the energy backup capability of plug-in hybrid electric vehicles (PHEVs), will be able to be used as an alternative energy source and also power the house in the event of a grid outage.

Zhou et al. [87] introduced a correlated Q-learning-based approach for microgrid energy management.

Zhou et al. [88] introduced a bayesian deep reinforcement learning method for the microgrid energy management under communication failure. This study assumes that the communication environment between microgrid elements is not perfect and, therefore, energy management systems must be able to maintain a consistent level of performance under these conditions.

Although various prediction and operation methods are proposed for HEMS, the relationship between load and solar generation prediction along with operation is not investigated, e.g., how different load/supply forecasting methods will affect the operation results. This dissertation differs from existing works in two aspects. First, an advanced forecasting method, namely sequence-to-sequence learning, is applied for a device-level load prediction using real-world data. Second, based on several baseline algorithms, we further investigate how the prediction accuracy will affect HEMS at the operation level.

Furthermore, most aforementioned works apply a perfect data collection and agent training environment, which is a strong assumption in reality. In the real-world environment, fine-grained and good-quality data may be inaccessible due to privacy concerns, measurement errors, and so on. As such, synthetic data-based HEMS is much more realistic since the agent can use pre-generated and fine-grained data for training. Finally, although there have been many advanced RL algorithms such as deep Q-learning [86], double deep Q-learning, and so on, here we select Q-learning because i) Q-learning is the most generally applied RL algorithm; ii) the HEMS is considered as an MDP with limited state and action space in this work, which makes Q-learning an ideal candidate.

2.3 Smart Home Synthetic Data Generation

A wide variety of methods have been developed for synthetic data generation of smart grid applications. In the literature on smart homes, the majority of the studies generate synthetic data by applying either model-based or data-driven approaches. Researchers in model-based approaches typically describe the features of household devices with hand-crafted features and mathematical equations.

2.3.1 Model-based Data Generation Methods

For example, [13] proposes a bottom-up residential building energy simulation, which simulates occupant behavior patterns with a Markov chain clustering algorithm, [18] develops a bottom-up analysis method for the residential building energy consumption by combining non-intrusive load decomposition and Markov chain methods. A statistical synthetic data generator is defined in [30] for electric vehicle load modeling, in which the Gaussian mixture model is used to estimate the connection time. [49] applies Gaussian Mixture Models (GMMs) to represent key EV charging metrics based on probability density functions and combining these GMMs produces realistic EV profiles. In addition, [38] defines a smart residential load simulator based on MATLAB-Simulink, and it includes dedicated physical models of various household devices.

2.3.2 Data-driven Data Generation Methods

On the other hand, considering the high complexity of the above-mentioned model-based methods, the data-driven methods become a favorable replacement as they do not require prior knowledge. Recent developments in using GANs have been successful in producing synthetic time-series data. [15] coupled non-intrusive load decomposition with conditional GAN to generate synthetic labeled

(e.g., appliances) load patterns and usage habits, which requires no model assumptions. Furthermore, for smart grid scenario generation, [12] introduces a model-free method for scenario generation of smart grid, and GAN is used to capture the spatial and temporal correlations of renewable power plants. Similarly, [16] applies GAN to features derived by ARIMA and Fourier transform for generating realistic energy consumption data by learning from actual data. [82] uses the GAN network to learn the level and pattern features of smart home time-series data, both of which are defined by household consumption and activity, and generate synthesized data with a distribution similar to the real data. [68] clusters daily load profiles with known clustering techniques and use a GAN network to synthesize daily loads for each cluster. [10] proposed a GAN network with Wasserstein loss function to learn temporal and power patterns of EV charging sessions and create a synthetic load dataset for electric vehicles.

As part of the effort to address limitations in real datasets, we observed in our previous works, such as the sequence-to-sequence learning-based method for load prediction in [50] and Q-learning based scheme for smart home energy management in [51], in the chapter 5 of this dissertation we proposed a novel VAE-GAN method for generating a realistic synthetic smart home dataset based on [52] and using this dataset to train a Q-learning model for smart home energy control. This is different from GAN-based approaches as a variational autoencoder network is deployed in the GAN generator, to overcome the mode collapse issue of the traditional GAN.

CHAPTER 3: RESIDENTIAL APPLIANCE-LEVEL SHORT-TERM LOAD FORECASTING

In this chapter, We propose an LSTM recurrent neural network-based model to predict home appliances' energy consumption for the upcoming hours[1]. After pre-processing the data collected from the smart meter, we feed it into a deep neural network to produce the predictive load consumption of the appliances. Figure. 3.1 displays the framework of the proposed model.

[1]The material presented in this chapter has been published at the 2020 IEEE Global Communications Conference (GLOBECOM) under the title "Residential Appliance-Level Load Forecasting with Deep Learning" by Mina Razghandi and Professor Damla Turgut.

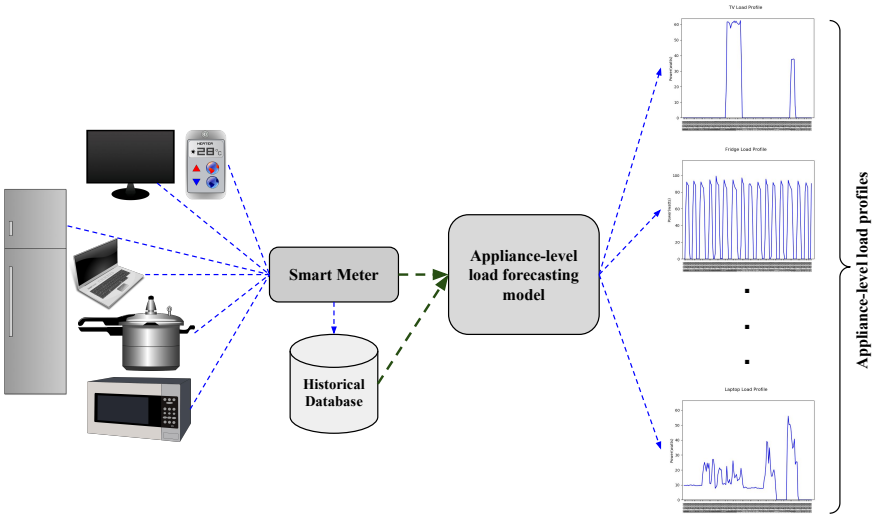


Figure 3.1: Framework for the proposed appliance-level load forecasting with the LSTM network.

3.1 Appliance-level Load Forecasting with Long Short-Term Memory

Recurrent neural networks are a form of feed-forward neural networks that rely on the previous stage output to produce new outputs. In other words, RNNs perform the same input calculation at each stage, taking into account feedback obtained from previous stage results, thereby generating associations with the input feature with its internal memory.

The problem we are investigating here is estimating future load consumption patterns for each home appliance having historical data and using multiple variants that affect output. We assume that the output is a continuous value depending on the previous value and other factors. We believe that LSTM recurrent neural networks are more suitable than standard feed-forward neural networks as LSTM can learn the long-term dependencies in the input features due to the memory cells built into its architecture design.

Table 3.1 displays the hyper-parameters used to train the LSTM model for predicting future appliance loads at 10-minutes intervals, with the strongest performance for the ones in bold text. When the LSTM model is trained, it can forecast the load consumption for a given appliance, hour, minute, day of the week, and other inputs. In general, the model can predict future energy usage for each home appliance at various time intervals for the following days over a week. The LSTM-based model architecture is displayed in Figure. 3.2.

Table 3.1: LSTM-based short-term load forecasting model hyper-parameters.

Parameters	**Value**
Number of layers	**2**, 3, 4
Learning Rate	**0.0001**, 0.001, 0.1
Number of hidden neurons	**128**, 256
Sequence length	**12**, 18, 36, 72, 144
Batch size	128, 512, 1024, 2048, **4096**
Optimizer	GSD, **Adam**
Interval	**10**, **15**

3.2 Evaluation Study

Here we will discuss the setup of our experiment, the dataset we used to evaluate the performance of our load forecasting model, the data preprocessing steps we used to prepare our dataset, the baseline models against which we measured the performance of our model, and finally our evaluation results and discussion of the results.

3.2.1 Experiment Setup

In the experiments, we have used a publicly available dataset, DRED [63], collected in a household from July to December 2015. DRED dataset provides details on the deployment of sensors that monitor aggregated and appliance level energy consumption of a typical household in the Netherlands, with a sampling frequency of 1Hz of 12 different home appliances. This dataset also includes ambient information (room-level temperature, outdoor temperature, environmental parameters (wind, humidity, precipitation), occupancy information (room-level location of occupants), and overall household information (layout, monitored appliances, location mapping and etc.). All this information can be very useful to build a strong model for households with similar characteristics. However, data availability was not consistent for other datasets except for energy consumption, so we could not integrate them into our prediction model. However, using only energy consumption historical data makes the model to be more generalizable, meaning with only historical energy consumption data for each household we can implement a predictive model.

Not many households monitor appliances' energy consumption using smart meters. But the predictive models are data-driven meaning that the only contributing factor is rich historical data on devices' energy consumption, no other assumptions about the number of residents, house square footage and etc are not considered. DRED dataset is a typical household, but the predictive models

require historical data for any individual house that we want to predict their energy consumption. We used the top six most energy-consuming appliances for our experiments: television, refrigerator, laptop computer, electric heater, microwave, cooker, and the overall house energy consumption. After data pre-processing, there are around 218880 rows of data (152 days) for each appliance. With 60%, 30%, and 10% ratios, we split data into three-part train, validation, and test sets.

We have implemented our model on an NVIDIA GeForce GTX 1070/PCIe/SSE2 with 15.6 GB memory and 3.6 GHz core clock hardware configuration and used Pytorch library and Python for software implementation.

We used two other load forecasting methods as the baseline for evaluating the performance of the proposed model:

- **Random Forest (RF)**: Most of the previous studies use ARIMA model [60] for load forecasting; however, ARIMA essentially implies a linear relationship between input variables that may be dependent or independent which is not always true for multivariate problems [25]. Random Forest [35] algorithm, on the other hand, has more promising results in a multivariate load forecasting problem.
- **FeedForward Neural Network (FNN)**: We implemented a basic feed-forward network with three fully connected layers to compare its results against the LSTM model (See Figure. 3.2). The FNN has three fully connected layers with a ReLU activation function [1] as the last layer. The fully connected layers have 128 hidden neurons each, the same as LSTM layers.
- **Long Short-Term Memory Neural Network (LSTM)**: As described in Section 3.1.

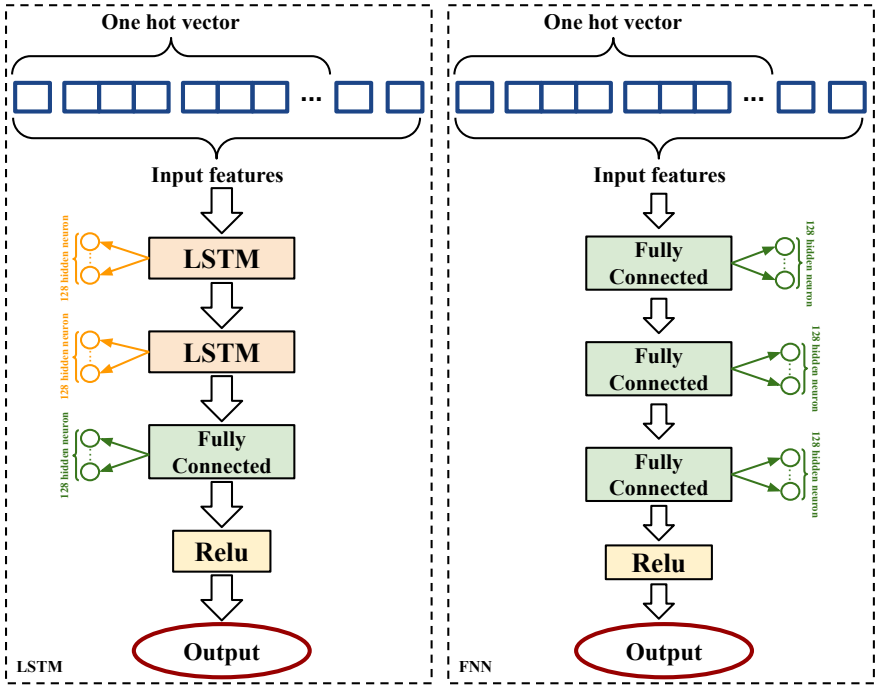


Figure 3.2: LSTM-based and FNN-based architectures used to predict appliance-level short-term load consumption.

3.2.2 Performance Measures

We used Root Mean Square Error (RMSE) (Equation 3.1) and Normalized Root Mean Square Error (NRMSE) (Equation 3.2) to compare the performance of all three models where n is the total number of samples, y_j is the target value and $\hat{y_j}$ is the predicted value, $max(y_j)$ and $min(y_j)$ refer to the maximum and minimum electrical usage recorded for the appliance j.

$$RMSE = \sqrt{(\frac{1}{n})\sum_{i=1}^{n}(y_j - \hat{y}_j)^2} \tag{3.1}$$

$$NRMSE = \frac{\sqrt{(\frac{1}{n})\sum_{i=1}^{n}(y_j - \hat{y}_j)^2}}{max(y_j) - min(y_j)} \tag{3.2}$$

The RMSE is an excellent metric to demonstrate how far the predictions are from the target value. Since we have different appliances and each has its spectrum of energy usage to compare the results against each other, we have computed the NRMSE metric which normalizes the RMSE values with the relative load spectrum.

3.2.3 Data Preprocessing

Different appliances may be in use in any residential household, and the consumer behavior of utilizing home appliances is highly reliant on the consumer lifestyle. Several appliances can be in service multiple times a day, while some may be off for an entire week. Therefore, to forecast the future energy consumption of household appliances, it is necessary to extract certain features indicating the likelihood of using an appliance at a given time.

Several papers use deep learning models or statistical methods to forecast a household's total energy consumption for an entire day; however, more accurate details would provide further insights into the occupants, and they would be more valuable for improved energy-saving plans. We aim to predict appliance-level load consumption at different time intervals throughout the day. This time interval can vary, like every minute, every ten minutes, every fifteen minutes, or every half hour. However, as we increase the time between periods, the more difficult it becomes to measure the energy usage for devices with a working time duration less than the period.

For this purpose following information would be helpful: (1) *hour* (2) *minute* (3) *day of the week* (4) *Last_Seen_On*: the last time step appliance was running (5) *Last_Seen_Off*: the last time step appliance was off. Considering that the data has date and time details, it is simple to split date information into features such as "hour," "minute," and "day of the week." We can save them as categorical vectors for making the training of the model more straightforward.

We normalize the energy utilization data for each home appliance within the spectrum of maximum use to minimum use of the same appliance to prevent large numbers or anomalies from impacting the network's learning process. The minimum and maximum energy consumption for each electrical device is unique to its specifications and differs from other appliances. Thus, if we scale it to the same scope for all appliances, the network's output range is identified and would ultimately produce more reliable results.

With pre-processing data, we have introduced two new features: *Last_Seen_On* and *Last_Seen_Off*, to make it simpler for the model to learn consumption patterns for appliances with less repeated energy consumption patterns. We observe that the energy consumption is at the lowest level, and it goes up as the residents of a home decide to use the appliance except for the appliances running continuously, such as the refrigerator, AC, and heater. These two features are measured according to the historical data at each time step and help the model construct a view of the trend of consumption.

Having energy usage information for each appliance per minute, we can predict future energy consumption at various time intervals. One solution is to consider the average energy used over the time interval for the input data, but it may shrink the dataset size. On the other hand, we know that the more data we have, the better the model can learn the behavioral patterns of using appliances. Instead of estimating average consumption over time intervals, we generate an input sequence, as shown in Figure. 3.3, having data with $\frac{1}{60}$ Hz frequency for an entire day, but generating an input sequence with $\frac{1}{60 \times time\_interval}$ Hz frequency ($\frac{1}{600}$ Hz for 10-minute time intervals). Figure. 3.3 displays each data point in a day by its time and also demonstrates that we can construct different input sequences with data points within two-time intervals, and we can have more input data for the training process that is not augmented or duplicated.

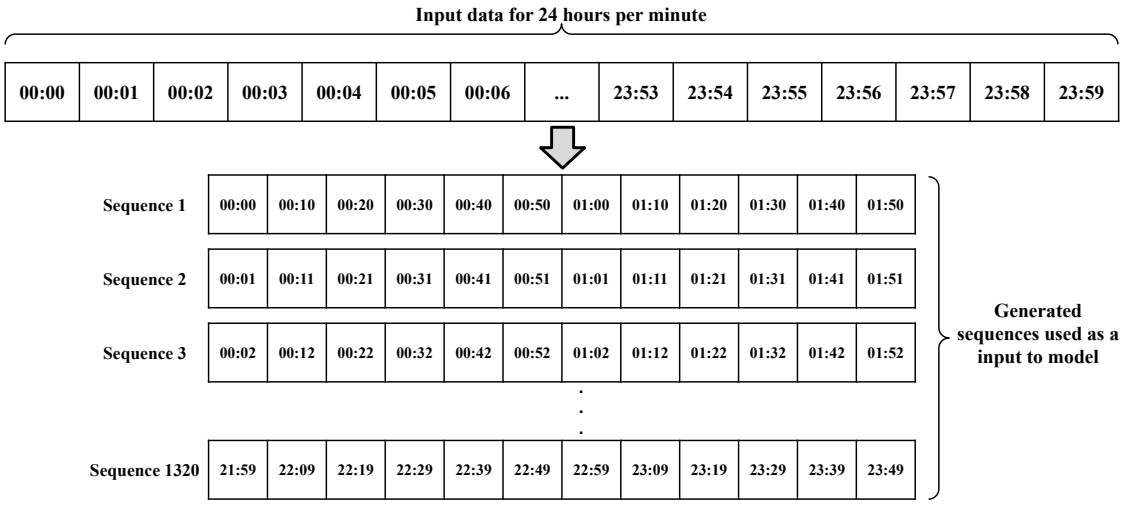


Figure 3.3: Generating input sequences for training with time interval = 10 minutes and sequence length = 12.

3.2.4 Evaluation Results and Discussion

In the first stage of our experiments, we explored the effect of using two newly introduced features, *Last_Seen_On* and *Last_Seen_Off*. To this end, we have trained our LSTM-based model on time, taking into account these two features and without considering these two features. The NRMSE metric was used to compare the forecasting results of the two models. Figure. 3.4 demonstrates that the NRMSE score for the LSTM-based model taking into account *Last_Seen_On* and *Last_Seen_Off* is considerably lower compared to the model that does not use these features specifically in appliance-level load forecasting.

Tables 3.2 and 3.3 present the outcomes of RMSE and NRMSE for the three load forecasting algorithms with ten and 15-minute intervals. The appliances used include a TV, refrigerator, laptop, electric heater, microwave, and cooker. We also show the total load.

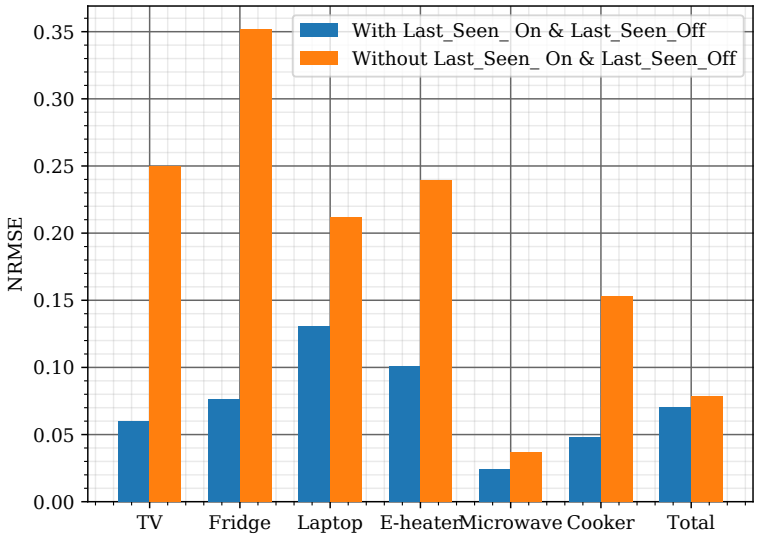


Figure 3.4: NRMSE results for LSTM-based short-term load forecasting model with and without taking into account *Last_Seen_Off* and *Last_Seen_On*.

The first conclusion we can draw from these numbers is that the LSTM network is a suitable architecture for this multi-variate load forecasting problem. The memory cells in the LSTM architecture help the network properly learn the long-term dependencies between the input variable, which regular feed-forward networks can not.

The other noticeable result is that for some appliances such as the laptop, the results of RMSE and NRMSE for FNN-based and LSTM-based models are quite close. As shown in Figure. 3.5.c, the LSTM-based model reaches the target value nearest in each step, while FNN and Random Forest models produce an average sequence value during the day to reduce the difference between target and predicted values and RMSE metric. However, for particular applications such as energy-saving scheduling, it is essential to know the precise load consumption at each time step nearest to the real value.

The last but not the least conclusion is that all three forecasting models can catch the total load

consumption pattern over the day (Figure. 3.5.g); however, the LSTM-based model is better at learning sudden changes.

Figures 3.5.a–3.5.g compare the forecasting performance of the three load forecasting models for TV, refrigerator, laptop, microwave, cooker, and total load consumption of the household. Actual energy consumption is indicated by the blue dashed line, LSTM-based predictive model results are indicated by the orange line, FNN-based predictive model results are indicated by the green line, and RF-based predictive model results are indicated by the yellow line.

In the example of watching television (Figure 3.5a), the LSTM model can observe the variance in energy consumption throughout the day and it can be assumed that this is a pattern that repeats for this household, such that the LSTM model can predict that pattern.

Refrigerators (Figure 3.5b) have very repeatable patterns of energy consumption, and all three models are able to follow that pattern, but only the LSTM-based model is able to hit the peak value and base value in every flux.

As a laptop (Figure 3.5c) can be used arbitrarily, a network needs to observe consumer behavior over time to be able to predict what will happen in the future, for example, when the laptop starts working, and as we can see here, the LSTM model is the most successful of the bunch.

During a typical day, the power consumption of the electrical heater (Figure 3.5d) fluctuates drastically depending on the temperature, etc, but we can see that, unlike the other two models, the LSTM model appears to hit each flux to almost its peak value and follow the pattern.

The problem with microwaves and cookers is that they are not used every day and if they are, they are used for short amounts of time. Also, their peak value is very high compared to their base value. Thus, it is very easy for a network to miss a flux or not predict the next step to go from zero to a very high value all of a sudden. Figures 3.5e and 3.5f demonstrate the success of the

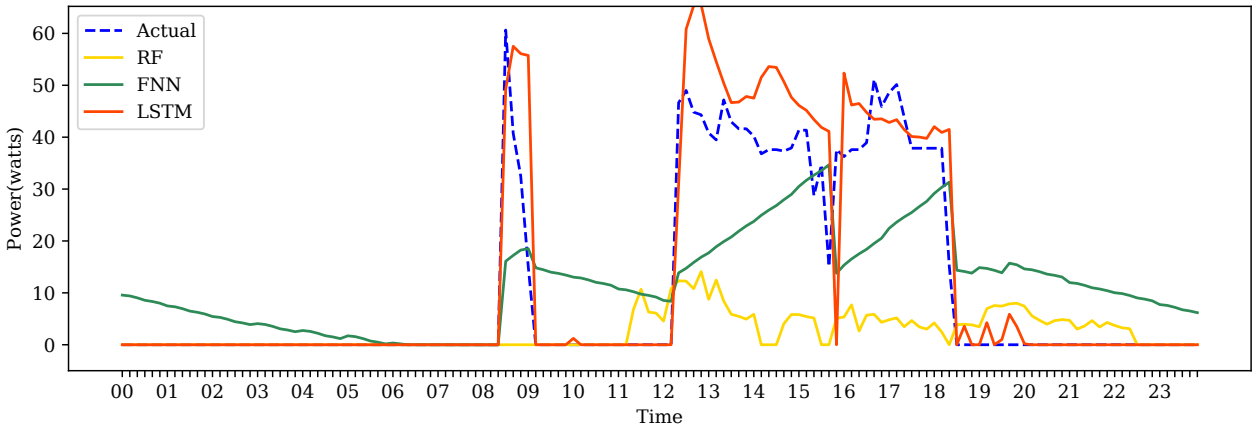
LSTM-based model in this regard.

Table 3.2: LSTM-based Short-term load forecasting performance evaluation Results for 10-minutes intervals

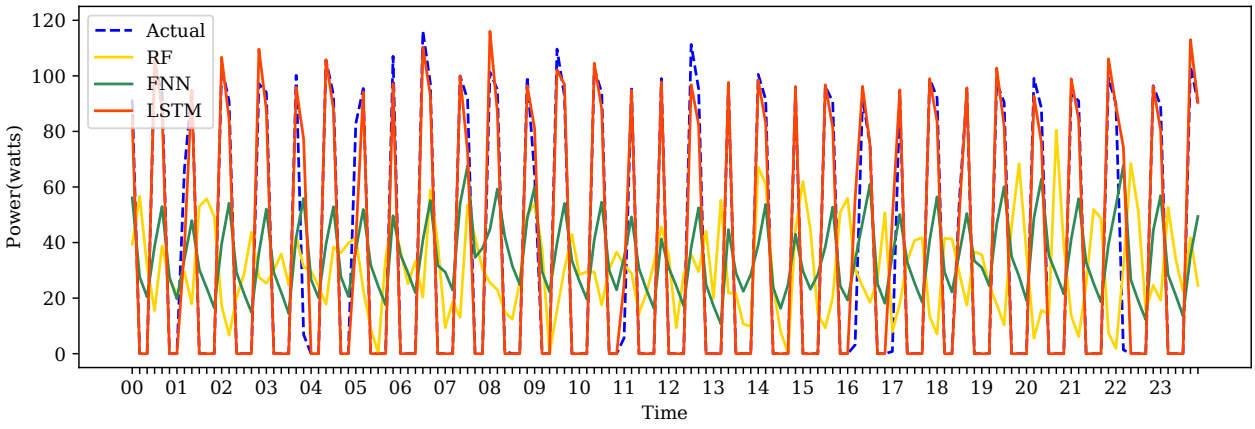
Appliance / Model	RF		FNN		LSTM	
	RMSE	NRMSE	RMSE	NRMSE	RMSE	NRMSE
TV	13.947	0.217	10.042	0.156	**3.849**	**0.060**
Refrigerator	47.775	0.319	33.755	0.226	**11.373**	**0.076**
Laptop	13.233	0.202	13.343	0.203	**8.575**	**0.131**
Electrical Heater	20.084	0.224	16.268	0.181	**9.091**	**0.101**
Microwave	57.649	0.042	64.149	0.046	**33.461**	**0.024**
Cooker	86.016	0.161	69.638	0.131	**25.648**	**0.048**
Total Energy Consumption	153.439	0.094	139.349	0.085	**128.665**	**0.079**

Table 3.3: LSTM-based Short-term load forecasting performance evaluation Results for 15-minutes intervals

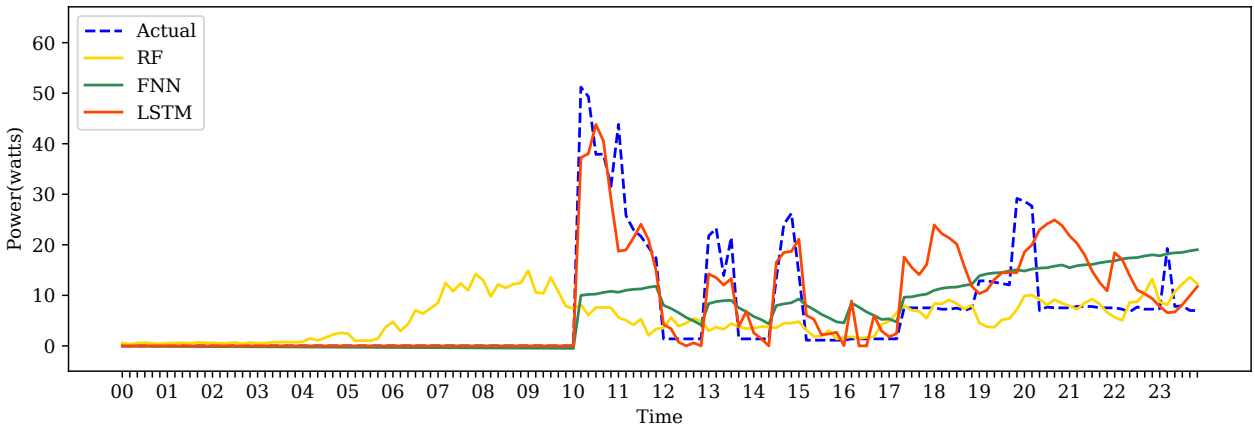
Model / Appliance	RF		FNN		LSTM	
	RMSE	NRMSE	RMSE	NRMSE	RMSE	NRMSE
TV	13.775	0.214	9.945	0.154	**4.214**	**0.065**
Refrigerator	46.275	0.331	34.094	0.244	**12.008**	**0.086**
Laptop	12.908	0.205	10.764	0.171	**9.627**	**0.153**
Electrical Heater	22.643	0.256	18.861	0.213	**9.786**	**0.111**
Microwave	60.748	0.043	81.402	0.057	**32.516**	**0.023**
Cooker	78.302	0.148	60.914	0.115	**33.036**	**0.062**
Total Energy Consumption	173.118	0.092	317.915	0.170	**152.543**	**0.081**



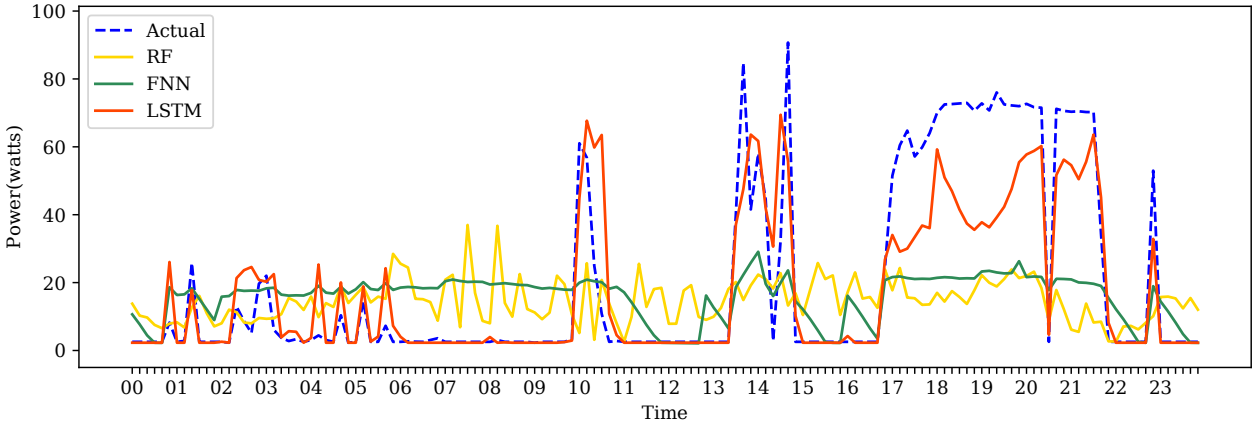
(a) TV



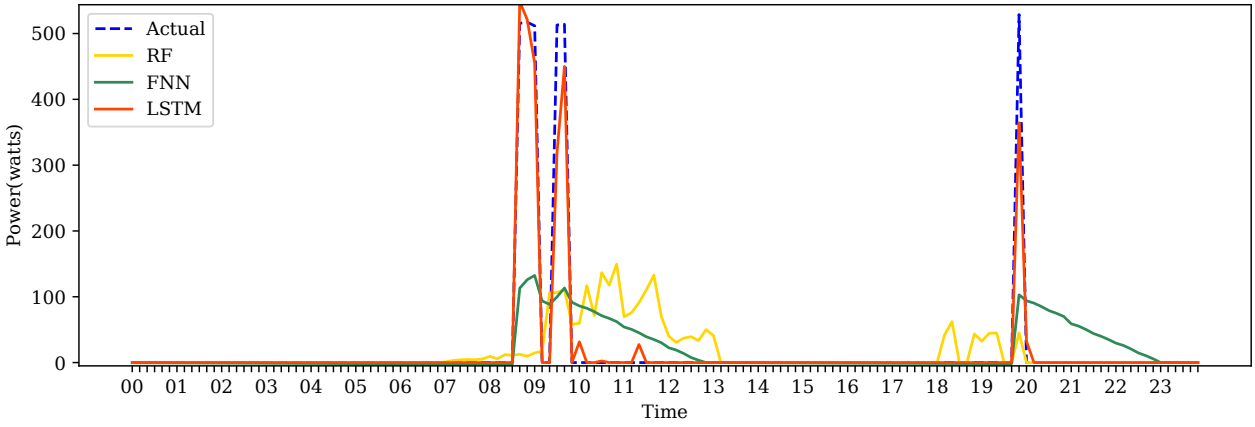
(b) Refrigerator



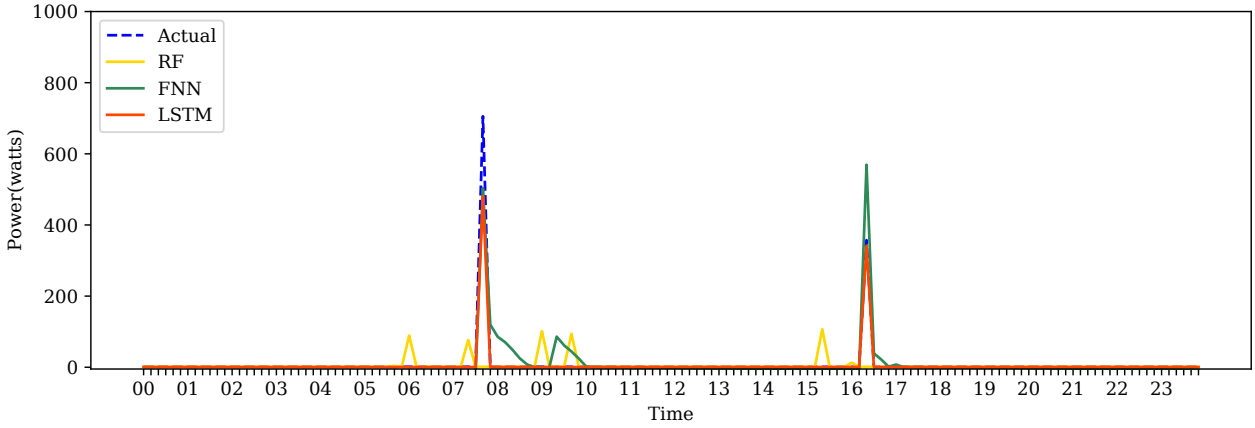
(c) Laptop



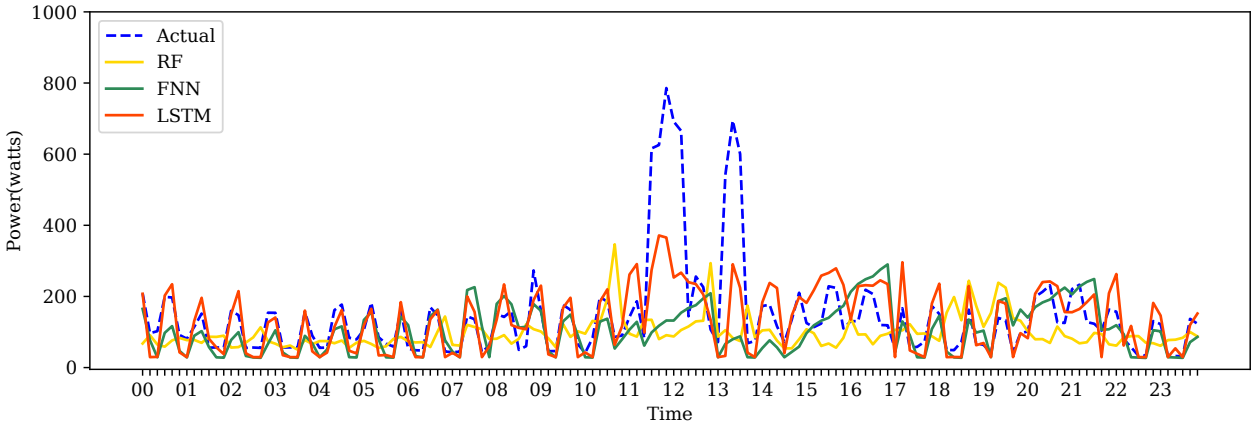
(d) Heater



(e) Cooker



(f) Microwave



(g) Total

Figure 3.5: Prediction of energy consumption for 24 hours with 10 minutes intervals based on LSTM-based short-term load forecasting.

CHAPTER 4: SHORT-TERM LOAD FORECASTING FOR SMART HOME ENERGY MANAGEMENT

In the first section of this chapter[1], we propose a Sequence to Sequence (Seq2Seq) encoder/decoder model to predict the energy consumption of smart home appliances which represents the behavioral pattern of smart home residents. The Seq2Seq encoder/decoder model derives occupants' habits from historical data and predicts hour-ahead appliance-level energy consumption in a household.

We implement a Bi-LSTM-based sequence to sequence learning model in the second section of this chapter[2] to improve network prediction results. We use this model to predict home appliances' load consumption and PV power. On the basis of the prediction results, we then optimize the HEMS offline using Q-learning. Finally, we examine the online performance of the trained Q-learning scheme with actual PV and load data.

4.1 Load Forecasting with Sequence To Sequence Learning

The following section outlines the proposed LSTM-based Seq2Seq learning load forecasting model. Afterward, we evaluate and discuss its performance in comparison with VARMA, dilated 1D convolution, and LSTM networks.

[1]The material presented in this section has been published at the 2021 IEEE International Conference on Communications (ICC) under the title "Short-Term Load Forecasting for Smart Home Appliances With Sequence to Sequence Learning" by Mina Razghandi, Hao Zhou, Professor Melike Erol-Kantarci, and Professor Damla Turgut.

[2]The material presented in this section has been published at the 2021 IEEE Global Communications Conference (GLOBECOM) under the title "Smart Home Energy Management: Sequence-to-Sequence Load Forecasting and Q-Learning" by Mina Razghandi, Hao Zhou, Professor Melike Erol-Kantarci, and Professor Damla Turgut.

4.1.1 LSTM-based Seq2Seq Learning Model

Sustkever et al. [59] first introduced the sequence to sequence learning technique in 2014. The idea was to encode a sequence of inputs to a fixed-length vector with a multilayered LSTM and then decode the fixed-length vector to a sequence of outputs with another multilayered, noting that input and output sequences length can vary. They used Seq2Seq learning to translate sentences from English to French.

We employed Seq2Seq learning to predict the energy consumption of smart home appliances for the next hour based on the previous 24 hours of historical data. As it is shown in Figure.4.1, our network architecture has three main LSTM network modules.

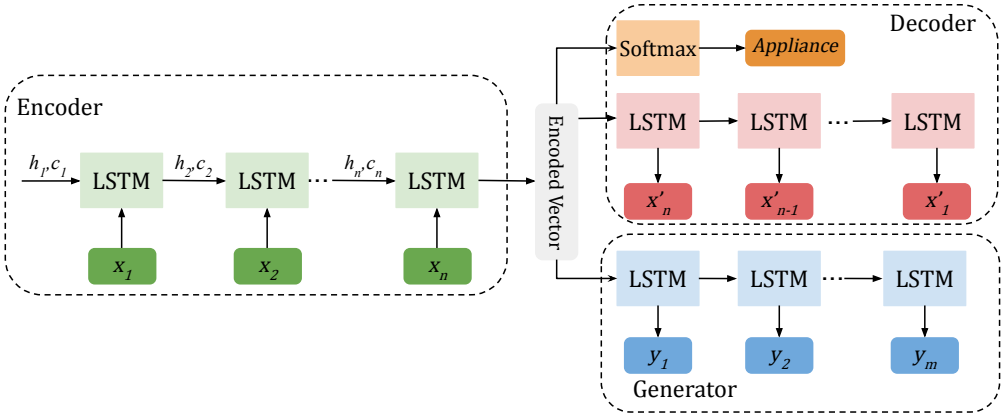


Figure 4.1: LSTM-based Seq2Seq encoder-decoder model.

Considering $(x_1, x_2, ..., x_n)$ as the input sequence for the encoder module, the LSTM **encoder** network maps the input sequence to a fixed-length vector Z, which is mainly based on the last hidden state of LSTM. The initial hidden state of the LSTM network in the **decoder** module is set to the fixed-length vector Z and its goal is to compute and maximize the conditional probability of sequence $(x_n^{'}, x_{n-1}^{'}, ..., x_1^{'})$ of $(x_1, x_2, ..., x_n)$:

$$p\left(x_n^{'}, ..., x_1^{'} | x_1, ..., x_n\right) = \prod_{t=1}^{n} p\left(x_{n-t}^{'} | x_n^{'}, ..., x_{n-t+1}^{'}, Z\right) \tag{4.1}$$

In the Equation 4.1, $(x_n^{'}, x_{n-1}^{'}, ..., x_1^{'})$ is the reversed form of input sequence $(x_1, x_2, ..., x_n)$ enforcing encoder network to obtain long term dependencies of input sequence with putting more weight on most recent ones by minimizing a weighted MSE loss function (Equation 4.2). Besides this, by applying a Softmax function to the fixed-length vector Z, the decoder module predicts the type of appliance that was not given in the input features.

$$WeightedMSE = \frac{1}{n} \sum_{i=1}^{n} W * (x_i^{'} - x_i)^2 \tag{4.2}$$

Ultimately, the LSTM network in the **generator** module, estimates the conditional probability $p\left(y_1, ..., y_m | x_1, ..., x_n\right)$ according to the Equation 4.3:

$$p\left(y_1, ..., y_m | x_1, ..., x_n\right) = \prod_{t=1}^{m} p\left(y_t | y_1, ..., y_{t-1}, Z\right) \tag{4.3}$$

In Equation 4.3, $(y_1, ..., y_m)$ is the prediction results for next m data points obtained from the fixed-length vector Z by minimizing a MSE loss function:

$$MSE = \frac{1}{m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 \tag{4.4}$$

4.1.2 Evaluation Study

The following sections describe how we conducted our experiments, the dataset used for experiments, the processes we used to analyze it, benchmark models, and performance comparison metrics. The results will be presented and discussed.

4.1.2.1 Experiments Setup

In the evaluation study, we used a publicly available dataset, GreenD [43], collected from eight different households, including apartments and detached houses, from December 2013 to October 2014 in Austria and Italy. Some research needs to be done to investigate whether we can use a trained model for predicting energy consumption for households with similar situations but in our experiments we use historical data for each household to train a predictive model and predict their energy consumption. But since the model is data-driven it can be extended to any household that has consistent energy consumption historical data. As it is examined by using the GreenD dataset. Using this dataset, we intend to demonstrate that our proposed forecasting model's predictions are consistent when applied to different households with different backgrounds, as it is not dependent on any one household. After data-processing, only four buildings had sufficient data for training a deep learning model (more than 150 days) with $\frac{1}{60}$ Hz frequency. This dataset contains appliance-level energy consumption for an average of nine appliances per household at 1Hz frequency. However, not all of the monitored appliances display a repeated pattern of the occupant's lifestyle. Therefore, Table 4.1 presents the five selected home appliances that were in common among all four buildings and had a repetitive trend. Artificial intelligence techniques are able to learn trends that have some kind of a repetitive pattern. In the concept of home appliances, the living habits of the residents and their regular daily routines introduce a consumption pattern for certain appliances such as TV, lamp, coffee machine, air conditioner, etc. Some might be in

use every day at certain hours, some might be in use weekly, and some seasonal. AI can learn these underlying relations from the historical data if enough data is provided and make predictions according to the learned features. But for some devices that are in use arbitrarily and do not fit into regular daily routines, it is a very difficult task to learn their behavior and make accurate predictions. With 70%, 20%, and 10% ratios, we split data into three training, validation, and test sets.

Table 4.1: The building numbers, dataset length, and monitored appliances used in the experiments.

Building Number	**Days**	**Monitored Appliances**
Building0	236	dishwasher, fridge, lamp, radio, washing machine
Building1	226	dishwasher, fridge, lamp, radio, washing machine
Building2	231	dishwasher, laptop, network-attached storage, TV, washing machine
Building3	151	computer, dishwasher, fridge, TV, washing machine

We used three load forecasting methods from the literature as the baseline for evaluating the performance of the proposed model:

- **Vector Auto Regression Moving Average (VARMA)**: This model is a combination of Vector Auto-Regression (VAR) and Vector Moving Average (VMA) models. VAR is a generalization of the Auto-Regressive model and is used when there are multiple parallel variables to forecast. VAR models the variables as a linear function of their past variables. The order of the VAR model determines the number of earlier time steps the model utilizes for predictions. VMA predicts the next steps based on a linear function of residual errors which is the difference between predicted and observed values to predict.

- **Dilated One Dimensional Convolutional**: This model is based on the WaveNet network, proposed in [47] that has dilated causal convolution layers to transform text to speech. The 1-dimensional convolution slides a filter on an input series usually by one stride, but then in the causal convolution, the ordering of the input data will remain the same, and the model

will not be learning based on future data. In a dilated convolution, the sliding filter skips input data with certain steps. Multiple stacked dilated convolutional layers allow larger input sequences but keep the network complexity efficient.

- **Long Short-Term Memory Neural Network**: LSTM network is a type of Recurrent Neural Network specifically designed to learn long-term dependencies in input data by relying on feedback from previous stage outputs. We used a 2-layer LSTM network in this paper.
- **Seq2Seq Learning**: As described in Section 4.1.1.

4.1.2.2 Performance Measures

We have used the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) metrics to evaluate regression prediction accuracy. However, since both RMSE and MAE are scale-dependent, we also have considered Normalized Root Mean Square Error (NRMSE) in our experiments. RMSE, MAE and NRMSE are shown in Equations 4.5–4.7 where n is the total number of samples, y_j is the target value and $\hat{y_j}$ is the predicted value, $max(y_j)$ and $min(y_j)$ refer to the maximum and minimum electrical usage recorded for the appliance j in the data, respectively.

$$RMSE = \sqrt{(\frac{1}{n})\sum_{i=1}^{n}(y_j - \hat{y_j})^2} \tag{4.5}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_j - \hat{y_j}| \tag{4.6}$$

$$NRMSE = \frac{RMSE}{max(y_j) - min(y_j)} \tag{4.7}$$

4.1.2.3 Data Reprocessing

Linear interpolation is used to complete missing data. Moreover, instead of using the minute-by-minute energy consumption of appliances, we smooth out the energy consumption of appliances over an interval of T=10 minutes, without losing generality. In our dataset, appliance energy consumption varies between 10W to 2000W; therefore, the values are normalized before being used in deep learning models.

Trends in energy consumption for electrical appliances can expose the living habits of household inhabitants. Some appliances would be used in a daily manner such as a TV or refrigerator, some others in a weekly fashion such as a vacuum cleaner, and some can display no specific usage patterns at all. It is worth noting that appliances with repetitive patterns of usability are more appropriate candidates to make predictions based on them. Accordingly, the historical data of power consumption and the day of the week are selected as the network input features. The row index will decide the minute and hour specification when the dataset is sorted in terms of time, so they are no longer needed as an additional feature input. Instead, for this model, the input sequence length of 144 points is chosen with T-minute intervals covering a period of 24 hours.

4.1.2.4 Evaluation Results and Discussion

Table 4.2 summarises RMSE, NRMSE, and MAE findings of four load forecasting algorithms for Building 0-3 with the lowest error in bold. Figures 4.2.a–f plot and compare the forecasting performance of all four prediction algorithms in a day snapshot against the ground truth for dishwasher, lamp, fridge, radio, laptop, and TV.

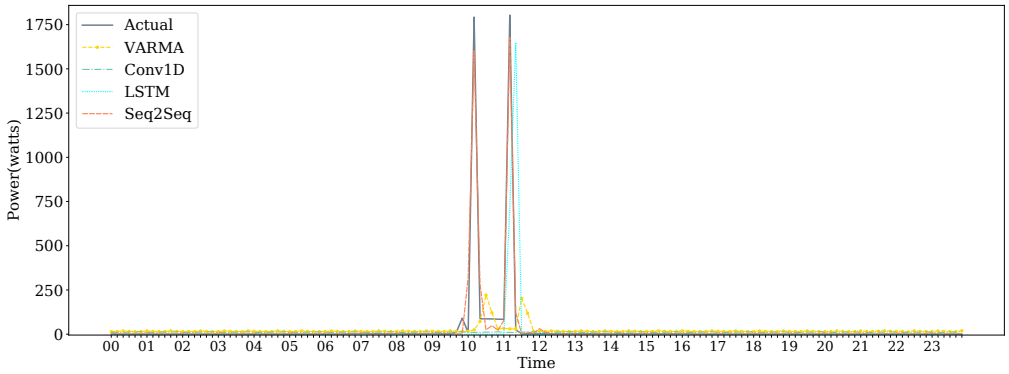
The results demonstrate that the LSTM-based Seq2Seq and LSTM models, respectively, outperform the other forecasting models and have noticeably lower prediction errors. LSTM model

architecture is quite similar to the Seq2Seq model architecture (encoder, decoder, and generator). Consequently, the LSTM model has the closest results to the Seq2Seq model, but the Seq2Seq model does a better job at learning the appliance working pattern and detecting appliance type because, unlike the LSTM model, the Seq2Seq model has no information about appliance type as an input.

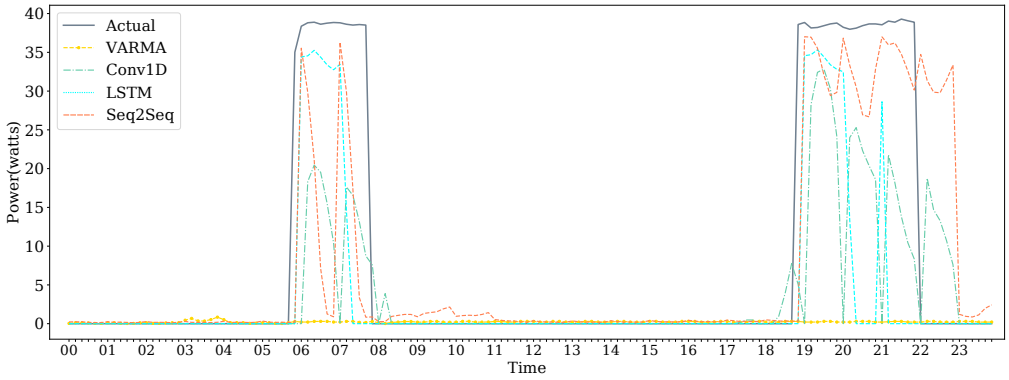
Note that, as seen in Figure. 4.2.c, all four prediction models were effective in learning patterns for the appliance with the most seasonal energy usage pattern (fridge), which means that these models are more promising when there is a recurring pattern of behaviors of smart home residents. However, the Seq2Seq model has better performance in load forecasting for appliances with a higher level of uncertainty in usage patterns.

An average dishwasher (Figure 4.2a) consumes 0-1750 watts when working and then another flux with the same features for a few minutes afterward. Across all the forecasting results, the Seq2Seq model was the only one that was able to predict both fluxes.

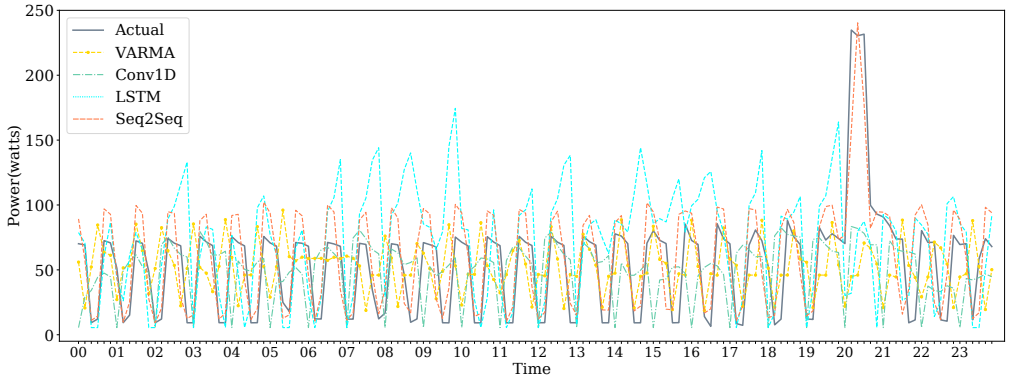
When considering daylight hours, the lamp's energy consumption (Figure 4.2b) is quite predictable; however, without having those details Seq2Seq can learn these features from historical data and produce better predictions as compared to other predictive models.



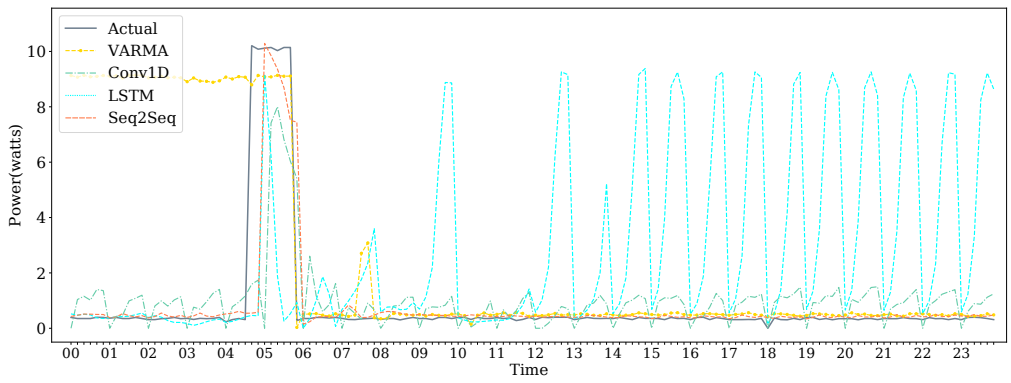
(a) Dish Washer



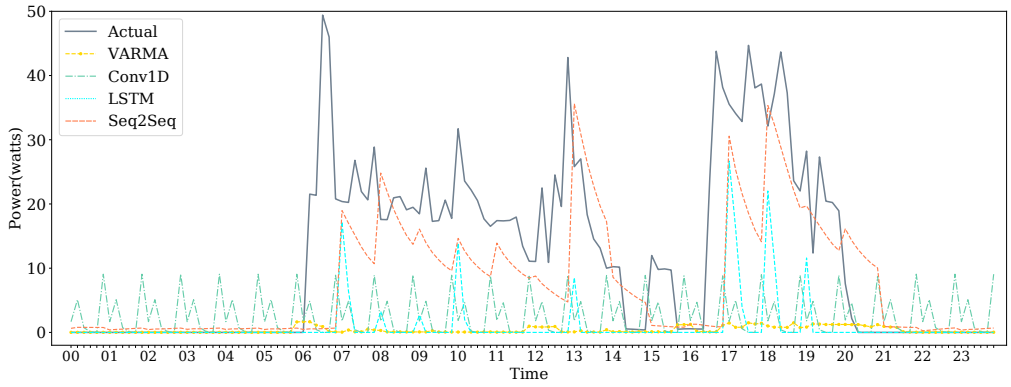
(b) Lamp



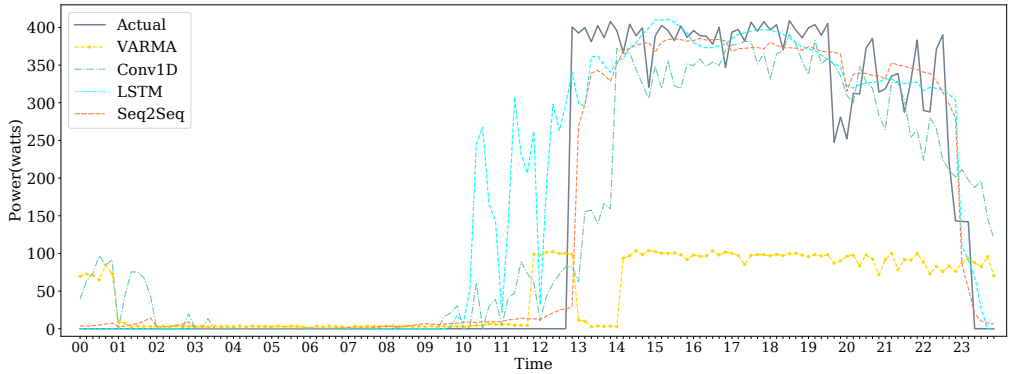
(c) Fridge



(d) Radio



(e) Laptop



(f) TV

Figure 4.2: Prediction of energy consumption for 24 hours with 10 minutes intervals based on LSTM-based Seq2Seq learning short-term load forecasting.

Table 4.2: LSTM-based Seq2Seq learning Short-term load forecasting performance evaluation Results for 10-minutes intervals.

Model	Building0														
	Dish Washer			Lamp			Fridge			Radio			Washing Machine		
	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE
VARMA	229.237	0.111	53.722	18.452	0.450	9.456	44.993	0.179	34.197	4.191	0.408	2.056	236.026	0.112	64.035
Conv1D	198.400	0.096	39.421	9.837	0.240	3.842	46.133	0.183	34.073	2.395	0.233	1.156	**214.037**	**0.102**	51.723
LSTM	493.948	0.239	168.236	11.632	0.284	3.942	46.601	0.185	33.218	3.513	0.342	1.805	246.735	0.117	**46.816**
Seq2Seq	**189.799**	**0.092**	**37.394**	**9.185**	**0.224**	**3.623**	**43.748**	**0.174**	**30.587**	**2.314**	**0.225**	**0.954**	215.298	**0.102**	48.373

	Building1														
	Dish Washer			Lamp			Fridge			Radio			Washing Machine		
	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE
VARMA	166.513	0.091	36.364	17.736	0.168	6.140	39.480	0.330	31.445	4.615	0.246	1.766	140.496	0.069	23.545
Conv1D	164.924	0.090	27.609	14.757	0.140	5.298	25.534	0.213	15.541	2.860	0.152	0.867	129.324	0.063	11.285
LSTM	154.280	0.084	18.174	**11.855**	**0.112**	**2.875**	36.070	0.301	20.999	**2.211**	**0.118**	**0.351**	120.479	0.059	33.571
Seq2Seq	**119.308**	**0.065**	**13.782**	12.803	0.121	3.420	**23.346**	**0.195**	**10.920**	**2.314**	**0.123**	0.420	**119.215**	**0.058**	**11.960**

	Building2														
	Dish Washer			TV			NAS			Laptop			Washing Machine		
	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE
VARMA	294.618	0.135	86.852	203.580	0.478	141.565	79.463	0.550	69.964	16.122	0.030	3.785	138.297	0.112	**23.292**
Conv1D	294.417	0.134	73.202	116.982	0.275	68.136	10.406	0.072	3.808	16.236	0.030	6.329	**137.647**	**0.111**	43.017
LSTM	299.399	0.137	**63.336**	124.385	0.292	61.287	**7.906**	**0.055**	**2.418**	15.800	0.029	3.364	177.153	0.143	98.461
Seq2Seq	**256.000**	**0.117**	69.832	**113.047**	**0.266**	**54.230**	9.307	0.064	2.450	**14.537**	**0.027**	**3.061**	142.829	0.115	34.219

	Building3														
	Dish Washer			TV			Fridge			Computer			Washing Machine		
	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE	RMSE	NRMSE	MAE
VARMA	157.059	0.088	29.454	34.862	0.363	14.512	70.842	0.558	49.705	38.798	0.433	25.994	189.403	0.095	39.776
Conv1D	152.446	0.085	16.818	22.188	0.231	8.625	41.075	0.323	24.815	15.416	0.172	8.242	165.717	0.083	**20.852**
LSTM	135.093	0.076	**12.546**	**18.148**	**0.189**	**4.666**	46.150	0.363	25.728	**14.639**	**0.164**	**5.947**	187.217	0.094	40.206
Seq2Seq	**131.433**	**0.073**	18.913	21.599	0.225	7.167	**32.821**	**0.258**	**17.385**	15.813	0.177	7.124	**164.082**	**0.082**	31.318

4.2 Q-Learning based Control Scheme with Seq2Seq Learning Load Prediction

In this section, inspired by the network proposed by Sustkever et al. [59], we used a sequence-to-sequence learning model for smart home appliances load forecasting. The load forecasting problem can be considered analogous to translating from one language to another in that the history window and the prediction window are not necessarily the same. Also, sequence-to-sequence learning has shown the potential to produce positive outcomes [50]. Additionally, we introduce a centralized HEMS model that uses the prediction methods of the previous section as inputs to make off-line optimizations.

4.2.1 Sequence-to-Sequence Encoder/Decoder Load Forecasting Model

Sustkever presented an end-to-end method for encoding an input sequence to a fixed-length vector and then decoding the vector to produce an output sequence, named sequence-to-sequence learning. They used a multilayered LSTM for both encoder and decoder networks. This network architecture is proposed with the primary goal of managing length differences between phrases while translating from one language to another, which deep neural networks cannot do because they require the dimensionality of the inputs and outputs to be specified and defined.

Let $(x_1, x_2, ..., x_n)$ be the input sequence fed into the encoder, v (fixed-length vector) the final hidden state of the LSTM encoder network, and $(y_1, y_2, ..., y_m)$ be the corresponding output sequence, where n, history window, and m, predicting window, may vary, the LSTM decoder network, takes v as the initial hidden state and computes the conditional probability of:

$$p\left(y_1, ..., y_m | x_1, ..., x_n\right) = \prod_{t=1}^{m} p\left(y_t | v, y_1, ..., y_{t-1}\right) \tag{4.8}$$

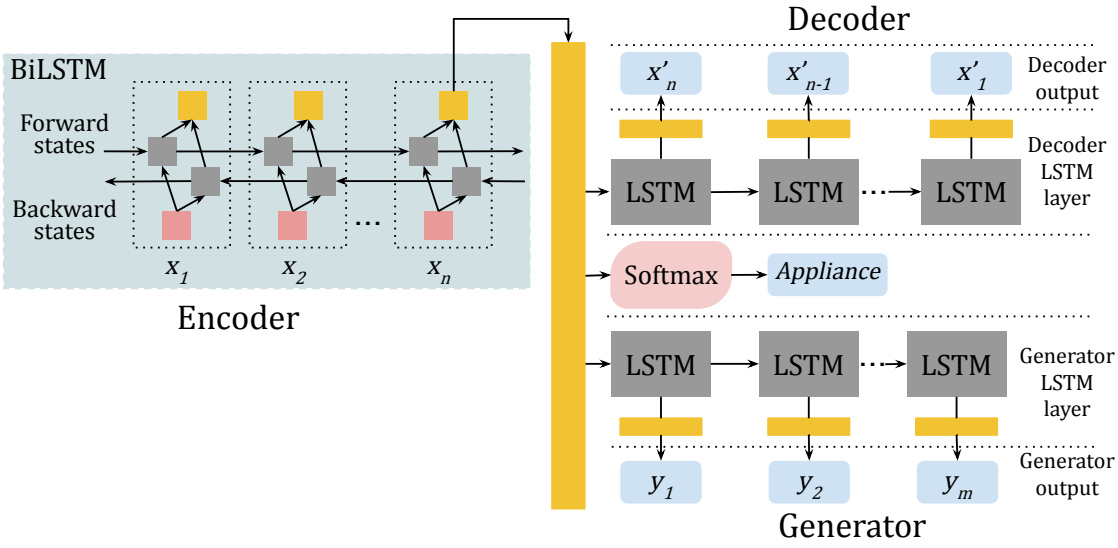


Figure 4.3: Sequence-to-sequence architecture: encoder (left) with one Bi-LSTM layer, decoder (upper right) and generator (bottom right) with one LSTM layer.

To perform load forecasting, the proposed model, as depicted in Figure. 4.3, uses the load demand history over the past 24 hours without specifying the type of appliance and predicts load demand for the hour ahead for each appliance. We have used a Bi-directional LSTM (Bi-LSTM) layer in our encoder module. Bi-LSTMs are a type of Bi-directional Recurrent network (Bi-RNN) [54] and benefit from using two independent LSTMs to read input data both in forwarding and backward directions. In the case of load forecasting, this architecture offers prominent perspectives for the network to understand and preserve underlying relationships from the past and future, for example, on-off time, and encode that information into the fixed-length vector. The decoder module's LSTM layer takes the fixed-length vector and regenerates the input sequence in reverse order, emphasizing recent data. Since the appliance type is not given as an input, the LSTM decoder predicts it, forcing the network to learn each appliance usage pattern. Finally, the LSTM layer in the generator module forecasts the load demand for the prediction window.

4.2.2 Q-learning based HEMS control

The HEMS in our research controls appliances' operation time without disturbing residents' convenience. There are several appliances in a household that are not time-sensitive and if we defer their operation time to off-peak hours considering residents' preferences we can reduce overall energy consumption. There is a defined waiting time for each appliance during which it may be delayed from operating. When it exceeds the waiting time, we expect consumers to get irritated. Historical data analysis was used to determine the wait times. As an example, we determined during which hours the dishwasher usually worked in the past and used those as the wait times. The optimization is measured according to the energy price during the operation time. The energy price is not consistent throughout the day.

The smart home consists of PV panels, loads, and an energy storage system (ESS). Based on the PV power and load prediction results, the controller aims to maximize the profit of smart home operations. The system is described as following:

$$\text{max} \quad \sum_{t=1}^{T}(P_t^E + P_t^{PV} - P_t^L)(\alpha p_t^{sell} + (1-\alpha)p_t^{buy}) \tag{4.9}$$

$$\text{s.t.} \quad \alpha = \mathbb{1}\{P_t^E + P_T^{PV} - P_t^L > 0\} \tag{4.9a}$$

$$P_t^E = P_{char} q_t \tag{4.9b}$$

$$S_{t+1} = S_t - \frac{P_t^E}{C^E} \tag{4.9c}$$

$$S_{min} \leq S_t \leq S_{max} \tag{4.9d}$$

$$P_t^L = \sum_{l=1}^{D} P_l G_{l,t} \tag{4.9e}$$

$$w_{l,t} \leq w_{l,max} \tag{4.9f}$$

where P_t^E, P_t^{PV}, and P_t^L denote the power of ESS, PV and load, p_t^{sell} and p_t^{buy} denote the price of selling and buying electricity, respectively. $\mathbb{1}$ is an indicator function. $\alpha = 1$ when $P_t^E + P_t^{PV} - P_t^D > 0$ to sell energy, otherwise $\alpha = 0$ to buy energy. P_{char} is the charging power of ESS, C^E is ESS capacity, $q_t = -1, 0, 1$ when charge, unchanged and discharge. Constraint (4.9c) is the ESS state of charge (SOC) updating constraints, and (4.9d) is the SOC upper and lower limit constraint. P_l is the power of home device l, D is the number of devices, and $G_{l,t}$ is the on/off status. In this work, we consider the demand side management, in which the controller can defer the operation time of some devices without affecting customer comfort. However, the devices have a waiting time limit shown as (4.9f), in which $w_{l,t}$ is the waiting time of device l, and $w_{l,max}$ is the max waiting time.

Algorithm 1 Q-learning for Smart Home Economic Dispatch

Prediction-based training:
Initialize: Q-learning and smart home parameters
Load Seq2Seq PV and load prediction results
for $episode = 1$ to E **do**
  for $t = 1$ to T **do**
    With probability ϵ choose action a randomly. Otherwise, $a = arg\ max(Q(s, a))$
    Agent calculates reward based on equation (4.9).
    Update agent state $\{t, S_t, w_{l,t}\}$ and Q-value:
    $Q(s, a) = (1 - \alpha)Q(s, a) + \alpha(r + \gamma maxQ(s', a'))$
  end for
end for
Get predicted reward
Test with actual data:
Load real PV and load data
Apply the trained Q-learning and get actual reward

Here, we use Q-learning-based control for HEMS operation. The formal definition of agent state is $\{t, S_t, w_{l,t}\}$, and the action is $\{q_t, G_{l,t}\}$. In this algorithm, we first perform prediction-based learning, which utilizes the prediction results of Seq2Seq learning. The trained action combinations are applied in actual PV and load data to verify the performance. The Q-learning-based algorithm is summarized in Algorithm 1.

4.2.3 Evaluation Study

We describe in the following section how we conducted our experiments, the dataset for training and testing the load forecasting model and Q-learning based HEMS, the approaches we used to analyze the dataset, benchmark parameters, and performance measures. We will discuss the forecasting accuracy and operational results.

4.2.3.1 Experiment Setup

We use a publicly available dataset from iHomeLab RAPT [22], which contains solar electricity production and domestic electricity consumption measurements for five houses located in Switzerland from April 2016 to August 2019 with a sampling frequency of five minutes. We focus on a residential-detached House D since it includes data on PV production, deferrable loads such as dishwashers, tumble dryers, High Fidelity (HiFi) systems and routers, washing machines, and total power consumption.

As a baseline for testing the performance of the proposed model, we used four load forecasting methods from the literature:

- **Vector Auto-Regressive Moving Average (VARMA)**: This model is a combination of Vector Auto-Regressive (VAR) and Vector Moving Average (VMA) models. The VAR model makes predictions assuming that the predicted value is a linear combination of past with random noise. The VMA model assumes the predicted value to be a linear combination of previous errors, expected value, and random noise term.
- **Support Vector Regression (SVR)**: SVR is a Support Vector Machine variant used to solve regression problems, where there is a nonlinear relationship between inputs. SVR will map

the input data to a higher dimension space in which it is linearly separable using kernel functions and fit a hyperplane with maximum margin to it.

- **Long Short-Term Memory Neural Network (LSTM)**: LSTM network is a variant of RNNs that aims to solve RNNs shortcomings for long data sequences. LSTM gating mechanism (input gate, forget gate, output gate) prevents exploding and vanishing gradient issues. Here we have used a two-layer LSTM.

- **Seq2Seq Learning**: This is described in Section 4.2.1.

We assume that the capacity of ESS in a smart home is 16 k·Wh, and the charge power is 4kW. The energy trading price is given in [86], where peak/off-peak pricing is applied. The Q-learning discount factor is 0.99, and the learning rate is 0.95. The initial ϵ value is 0.1, and it decreases with iterations. The prediction-based training contains 7000 iterations, and then it is applied to actual data.

4.2.3.2 Performance Measures

We picked three of the most common error measures in the literature to determine the accuracy of forecasts produced by the proposed Seq2Seq forecasting model and the other baseline models. The Weighted Mean Absolute Percentage Error (wMAPE) is defined as Equation 4.10 and is a variant of Mean Absolute Percentage Error that overcomes the divide by zero that occurs in MAPE when the base is zero. Another commonly used point error measure that is scale-dependent is Root Mean Square Error (RMSE). We have also used Normalized Root Mean Square Error (nRMSE) to compare the models' performance across different appliances with different scales. RMSE and nRMSE are defined according to Equations 4.11–4.12, where n is the total number of predictions, y_j is the actual value and $\hat{y_j}$ is the predicted value, $max(y_j)$ and $min(y_j)$ refer to the maximum and

minimum energy consumption recorded for the appliance j in the data, respectively.

$$wMAPE = \frac{\sum_{i=1}^{n}|y_j - \hat{y_j}|}{\sum_{i=1}^{n}|y_j|} \tag{4.10}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_j - \hat{y_j})^2} \tag{4.11}$$

$$nRMSE = \frac{RMSE}{max(y_j) - min(y_j)} \tag{4.12}$$

4.2.3.3 Data Preprocessing

The dataset contains energy consumption data with a sampling frequency of five minutes. We smooth out the energy consumption of appliances over T=10 minutes without compromising generality. The selected house has about 594 days of data after data processing. We divide data into training, validation, and test sets using a 70%, 20%, and 10% ratio. Since each appliance has its energy consumption fluctuation range, spanning from 0W to 15000W, it is critical to normalize data before feeding it to predictor models for the best performance.

Historical energy consumption data of a household is a rich source of knowledge that reveals profound insights into the activities and preferences of its inhabitants, which is valuable to HEMS systems. Hence, we used a sequence of 144 data points (from the previous 24 hours at 10-minute intervals) as input data, along with the day of the week.

4.2.3.4 Prediction Evaluation Results and Discussion

RMSE, nRMSE, and wMAPE are calculated based on the prediction results of the baseline models and presented in Tables 4.3–4.5. Figures 4.4.a–e demonstrates load predictions versus actual load demand for HiFi system and routers, dishwasher, PV power production, tumble dryer, washing machine, and overall energy consumption during a typical day using VARMA, SVR, LSTM and Seq2Seq models. Forecast results illustrate that the Seq2Seq model has the lowest error rate among all four models; in other words, it provides closer predictions to the actual demand. However, the RMSE and nRMSE of the LSTM model for tumble dryer and washing machine are 2.5% and 0.5% lower than the Seq2Seq model, which is insignificant considering the fact that the Seq2Seq has lower wMAPE for both mentioned appliances.

4.2.3.5 Operation Evaluation Results and Discussion

In this section, we will show the HEMS operation results under different prediction methods. Based on the PV and load prediction results of each day, Algorithm 1 produces both predicted and actual daily profit. This simulation is implemented in MATLAB, and each day is repeated for ten runs to achieve the average value. Figure. 4.5 shows the 40 days' prediction-based results and actual data test results of Seq2Seq and LSTM. The prediction-based results and actual data test results of Seq2Seq are close, while the prediction-based profit of LSTM is relatively lower than the actual data test results.

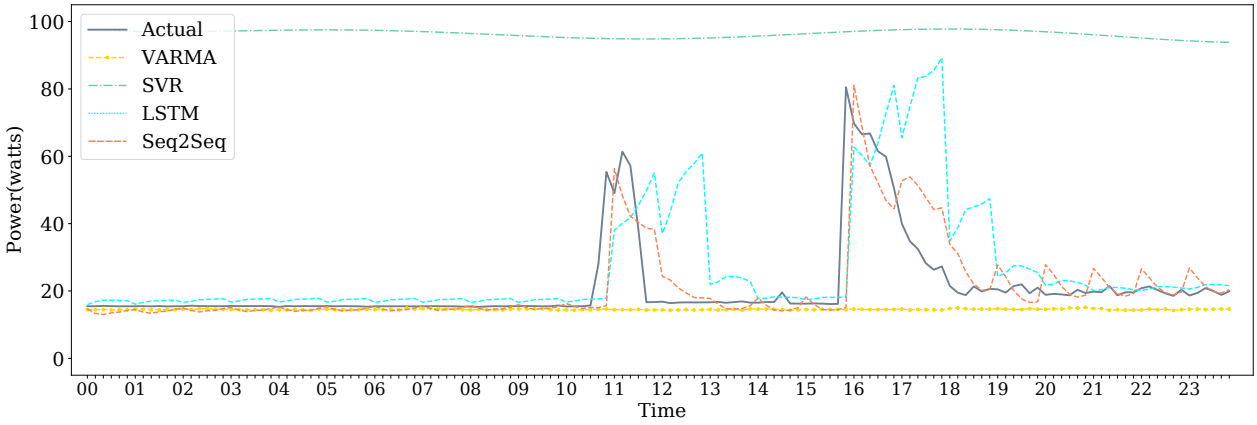
Figure. 4.6 presents the performance of VARMA and SVR, in which both algorithms have a noticeable error between prediction-based profit and actual data test profit. This error can be explained by Figure. 4.4c, where the VARMA and SVR have a much lower predicted power for PV. The lower predicted power naturally leads to lower prediction-based results. However, in the actual

Table 4.3: RMSE results of VARMA, SVR, LSTM, and Seq2Seq prediction models for each appliance. The bold indicates the best performance.

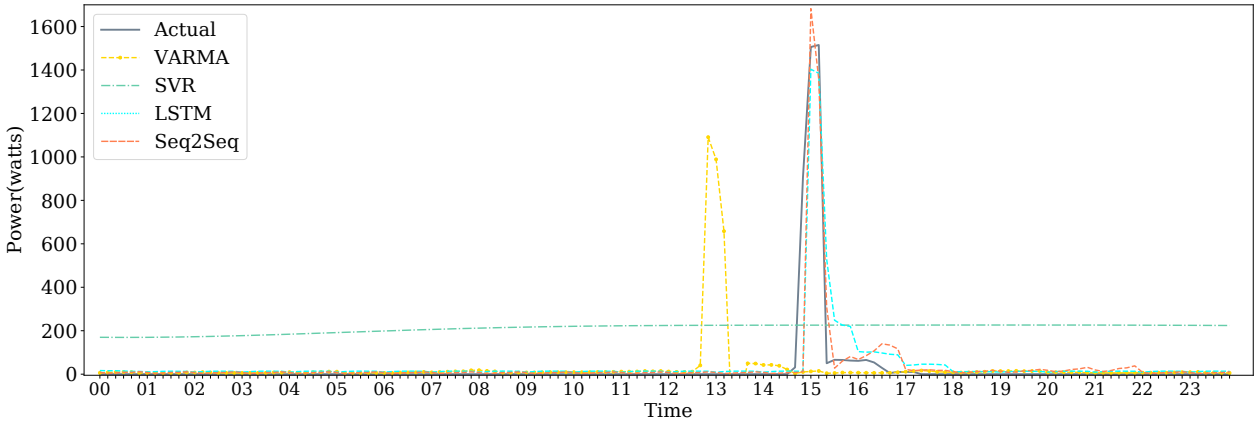
Appliance / Model	VARMA	SVR	LSTM	Seq2Seq
HiFi System and Router	31.53	65.50	12.38	**11.58**
Dishwasher	203.90	248.93	162.24	**159.99**
PV Power Production	1933.6	2086.1	879.65	**683.69**
Tumble Dryer	97.73	90.92	**53.50**	54.88
Washing Machine	198.92	232.16	**149.41**	150.25
Total Power Consumption	1588.1	1444.6	1025.7	**1017.2**

data test period, the actual PV power is much higher, and a high profit is observed. Figure. 4.7 presents the actual data test results of different forecasting methods, which is the primary concern of consumers. We also used the actual data for Q-learning-based HEMS as an optimal baseline, which means a prediction method with no error. Seq2Seq and LSTM-based HEMS benefit from the high forecasting accuracy, and a higher profit is observed. The Seq2Seq learning method has a comparable profit curve with an optimal baseline, while VARMA and SVR show a lower overall profit. On days 6, 28, 30, and 37, all algorithms have a comparable performance. A very low PV generation can explain those days, leading to a narrow profit difference in actual operation.

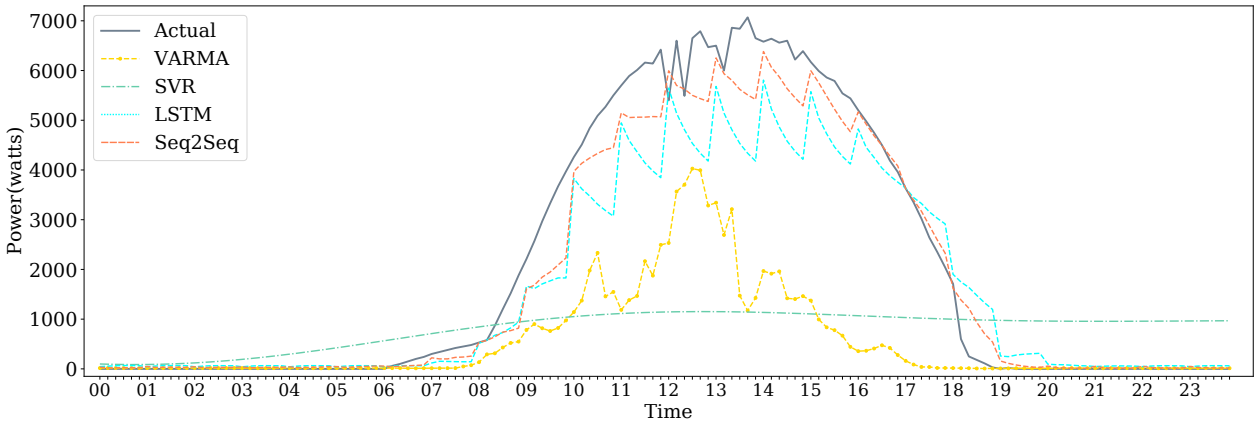
In summary, the Seq2Seq learning outperforms other algorithms with actual data test performance and a narrow error between prediction-based results. LSTM also results in good operation performance. VARMA and SVR have a significant error between prediction-based and actual data results.



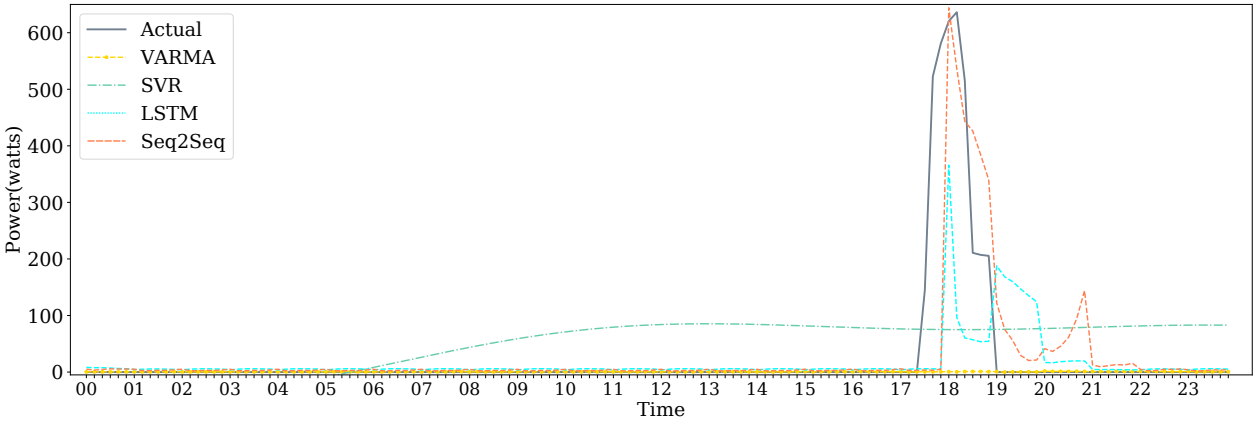
(a) HiFi System and Routers



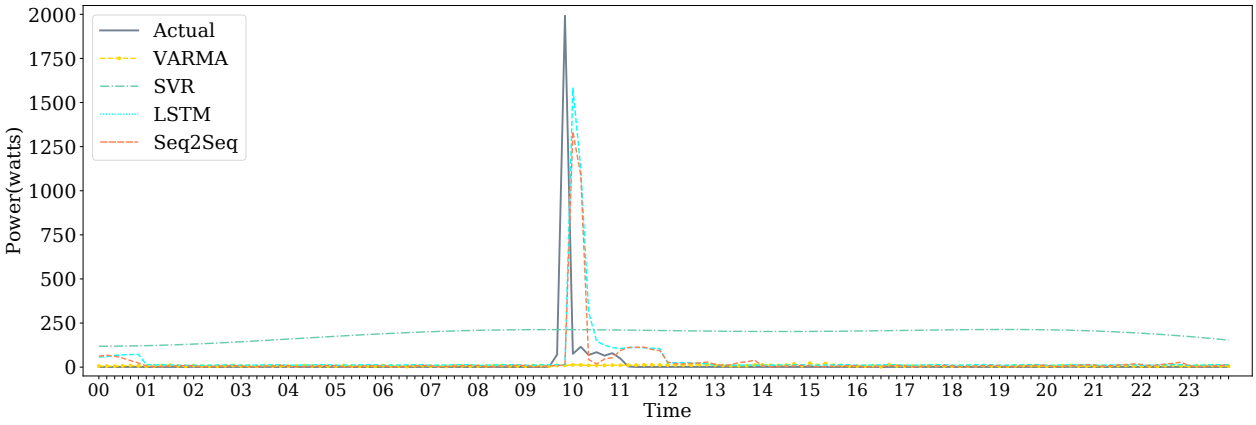
(b) Dishwasher



(c) PV Power Production



(d) Tumble Dryer



(e) Washing Machine

Figure 4.4: Prediction of energy consumption for 24 hours with 10 minutes intervals based on Bi-LSTM-based Seq2Seq learning short-term load forecasting.

Table 4.4: nRMSE results of VARMA, SVR, LSTM, and Seq2Seq prediction models for each appliance. The bold indicates the best performance.

Appliance / Model	VARMA	SVR	LSTM	Seq2Seq
HiFi System and Router	0.125	0.259	0.049	**0.045**
Dishwasher	0.106	0.129	0.084	**0.083**
PV Power Production	0.187	0.202	0.085	**0.066**
Tumble Dryer	0.117	0.109	**0.064**	0.066
Washing Machine	0.097	0.113	**0.072**	0.073
Total Power Consumption	0.133	0.121	0.085	**0.085**

Table 4.5: wMAPE results of VARMA, SVR, LSTM, and Seq2Seq prediction models for each appliance. The bold indicates the best performance.

Appliance / Model	VARMA	SVR	LSTM	Seq2Seq
HiFi System and Router	0.431	28007.08	0.263	**0.187**
Dishwasher	1.965	74463.87	1.579	**1.286**
PV Power Production	0.816	10238.64	0.378	**0.278**
Tumble Dryer	1.760	51988.42	1.216	**1.058**
Washing Machine	2.350	103436.34	1.743	**1.418**
Total Power Consumption	0.952	10240.58	0.546	**0.492**

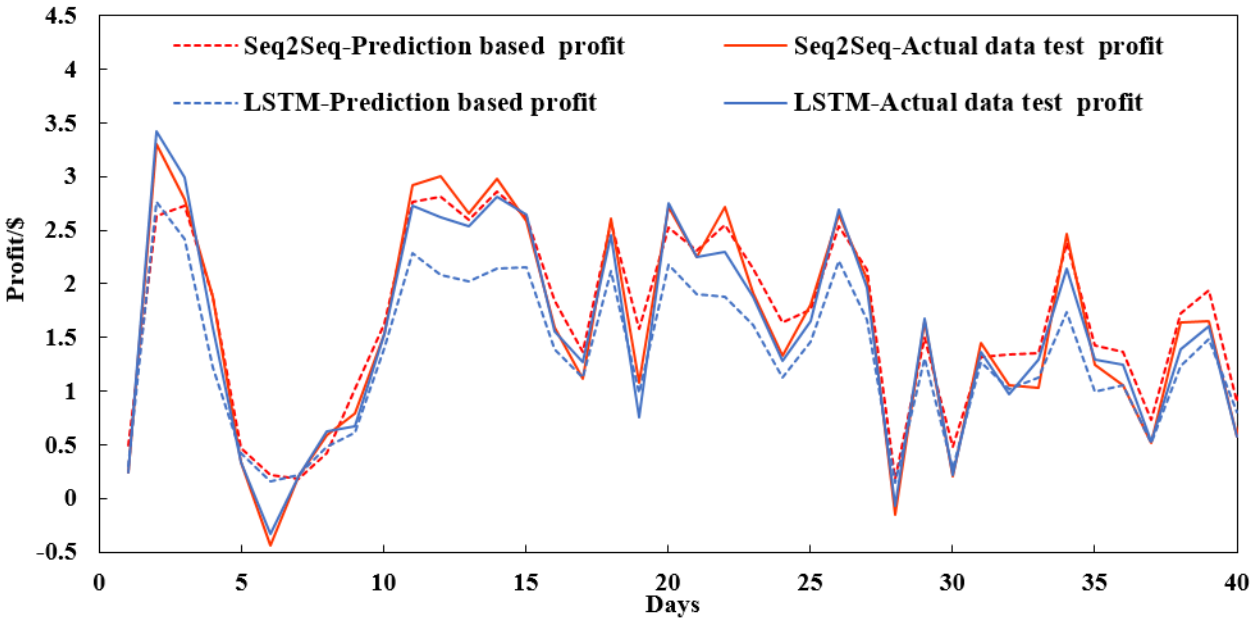


Figure 4.5: Operation comparison of LSTM and Seq2Seq.

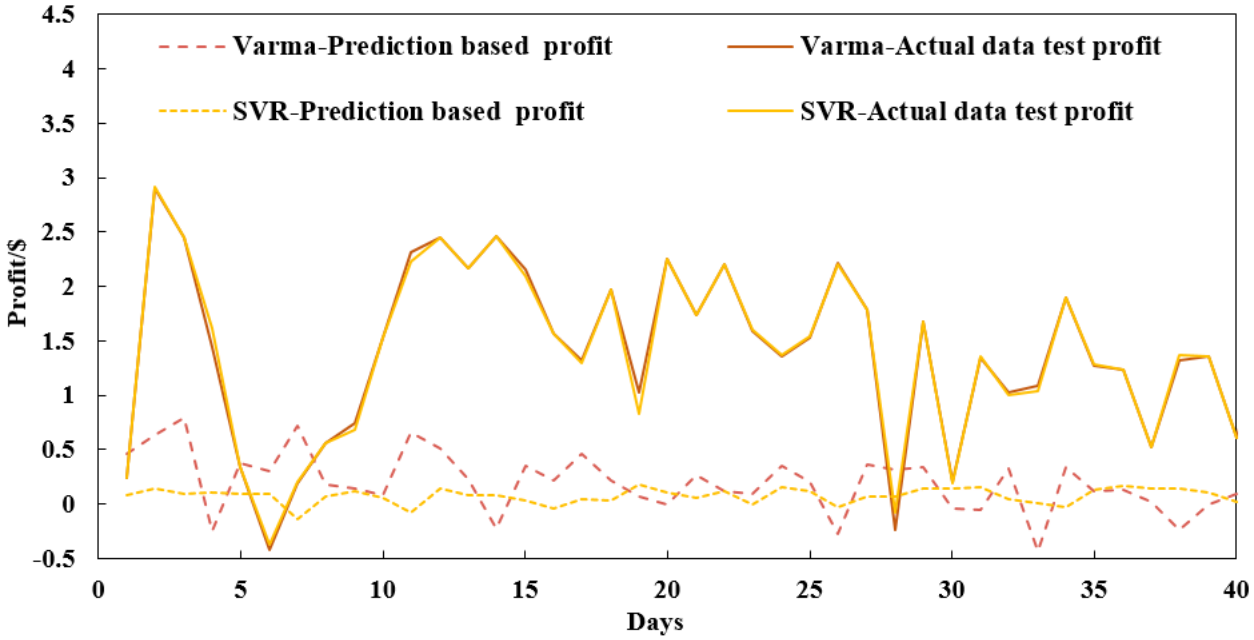


Figure 4.6: Operation comparison of VARMA and SVR.

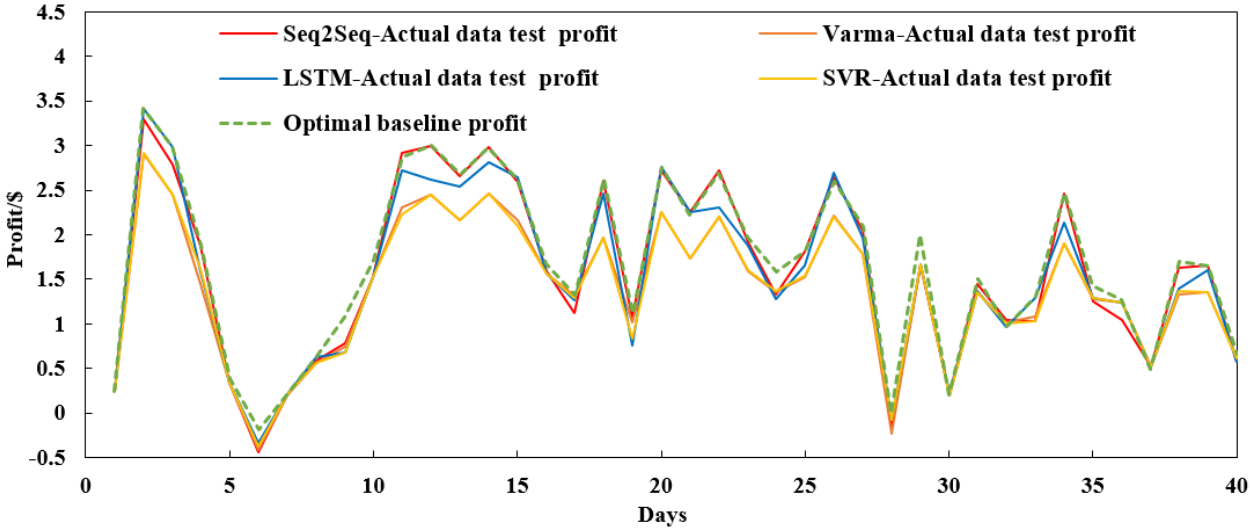


Figure 4.7: 40 days actual data test results comparison.

CHAPTER 5: SYNTHETIC DATA GENERATION FOR SMART HOME ENERGY MANAGEMENT SYSTEM

In the first section of this chapter[1], we adopt a Variational Autoencoder-Generative Adversarial Network (VAE-GAN) as a data-driven approach to generate realistic smart home data. VAE-GAN is used to generate daily overall electricity consumption and PV production data. As a foundation for the proposed model, this chapter also discusses well-known deep-learning-based generative models.

We continue our work on the VAE-GAN based synthetic smart home data generator in the second section of this chapter[2] to observe and preserve the temporal order of fluctuations in as well as the distribution of energy consumption in the historical data. Using this approach, we construct synthetic datasets of smart home overall load consumption, PV power production, and electric vehicle charging power consumption. In addition, we optimize Q-learning-based HEMS offline using synthetically generated datasets.

5.1 Smart Home Synthetic Data Generation

Following are sections aimed at explaining and describing VAE-GAN networks based on the basic and well-known deep learning models. Afterward, we will discuss the experimental setup, and performance metrics used for evaluating model efficiency and performance, and finally, we will

[1]The material presented in this section has been published at the 2022 IEEE International Conference on Communications (ICC) under the title "Variational Autoencoder Generative Adversarial Network for Synthetic Data Generation in Smart Home" by Mina Razghandi, Hao Zhou, Professor Melike Erol-Kantarci and Professor Damla Turgut.

[2]The material presented in this section is under review in 2022 IEEE Transactions on Smart Grid journal under the title "Smart Home Energy Management: VAE-GAN synthetic dataset generator and Q-learning" by Mina Razghandi, Hao Zhou, Professor Melike Erol-Kantarci and Professor Damla Turgut.

discuss the final evaluation results.

5.1.1 Data-driven Generative Models

Deep learning-based generative models, which use unsupervised learning to learn data distribution and underlying patterns, have gotten a lot of attention in recent years. Smart grid data, such as energy consumption, has greater complexity rather than repetitive seasonal data because it is tightly intertwined with residents' consumption behavior or environmental factors. Neural networks are powerful tools to overcome this complexity.

Among the well-known deep learning-based generative models, GAN and Variational Autoencoders are two of the best known. We will break down both of these models, as well as the VAE-GAN generative model and Q-learning based HEMS control, hereafter.

5.1.1.1 Autoencoder (AE)

An autoencoder is a neural network architecture that uses an unsupervised learning technique to compress the input dimensions into a compressed knowledge representation, called latent space, and then reverses the compressed knowledge representation into the original input dimensions. This architecture has two essential modules: encoder and decoder. The encoder modules map the input sequence (x) into the meaningful latent space (z), and based on z the decoder module outputs a reconstruction of the original input sequence ($\hat{x}$). The original input is unlabeled, but the decoder is trained to make reconstructions as close as the original input by minimizing the supervised reconstruction error, $\mathcal{L}_{reconstr}$, which is the distance between the original input and the subsequent reconstruction.

$$\mathcal{L}_{reconstr} = \|\hat{x} - x\|^2 \tag{5.1}$$

5.1.1.2 Variational Autoencoder (VAE)

After training the autoencoder model, the decoder module can produce new content given a random latent space. However, vanilla autoencoders suffer from a lack of regularity in the latent space, which means the latent space may not be continuous to interpolate for data points that are not present in the input sequence. Moreover, unregulated latent space can lead to severe overfitting.

Latent space regularity is reliant on the distribution of the input sequence and the dimension of the latent space, hence it is not always guaranteed. Variational autoencoders overcome this shortcoming by adding a regularization parameter, Kullback–Leibler (KL) divergence (Equation 5.2), in the training process, to ensure the latent space follows a Gaussian distribution and avoid overfitting. Instead of mapping the input sequence (x) to a vector, the VAE encoder (E) maps the data to two different vectors that are the mean and standard deviation parameters of a Gaussian distribution. Ultimately, by minimizing the $\mathcal{L}_{prior}$ loss, the encoder network is forced to compress the input sequence into a Gaussian distribution. In addition, it helps the decoder with reconstruction robustness, since the decoder module samples from a continuous distribution. The decoder loss is computed based on the distance between the reconstructed sequence ($\hat{x}$) and x. In the training process, the VAE minimizes a loss function ($\mathcal{L}_{VAE}$) that consists of two terms: the reconstruction term and the regularization term, which is the KL divergence between the latent space distribution and standard Gaussian distribution. $\mathcal{L}_{prior}$ and $\mathcal{L}_{VAE}$ are backpropagated through the network to train the VAE parameters.

$$D_{KL}(p\|q) = \sum_{i=1}^{N} p(x_i) log(\frac{p(x_i)}{q(y_i)}) \tag{5.2}$$

$$\mathcal{L}_{prior} = D_{KL}(E(x)||\mathcal{N}(0,1)) \tag{5.3}$$

$$\mathcal{L}_{VAE} = \mathcal{L}_{prior} + \mathcal{L}_{reconstr} \tag{5.4}$$

5.1.1.3 Generative Adversarial Network (GAN)

Generative models, such as GAN networks, are used to generate new data samples. GAN networks employ an unsupervised learning method to detect and learn patterns in input data and produce new samples that have the same distribution as the original dataset. This architecture consists of two neural networks: one which attempts to detect and learn patterns in the input data and produce a new sample, and the other which uses an unsupervised learning technique to distinguish between real input data and the synthesized sample. These two networks play against each other and actively seek an equilibrium.

- **Generator (G):** It maps a prior probability distribution, that is defined on input noise $p_z(Z)$, to a data space $G(z;\theta_g)$, where z is the input noise, and θ_g is the network parameter.
- **Discriminator (D):** $D(x;\theta_d)$ returns the likelihood of x being classified as a member of the original data, where 0 represents fake data and 1 represents original data.

G and D play an adversarial game shown by Equation 5.5, where D maximizes the probability of assigning true labels, $logD(x)$, and G tries to minimize the same probability, in other words, G attempts to mislead the D and maximize the final classification error, while D is trained to classify fake data more accurately and minimize the error:

$$\min_{G} \max_{D} \mathcal{L}_{GAN}(D,G) = E_x[log(D(x))] + E_z[1 - log(D(G(z)))] \tag{5.5}$$

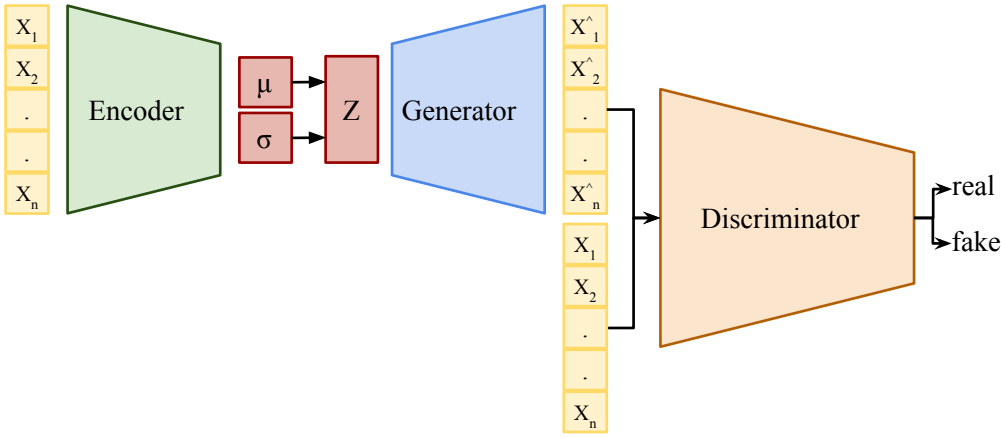


Figure 5.1: VAE-GAN model architecture. In this network, the encoder module maps the input sequence to the mean and the variance of a latent space with Gaussian distribution. The generator module reconstructs the input sequence from the latent space and tries to mislead the discriminator module to discriminate the generated sequence as a real sample. The discriminator module learns the distribution difference between real and fake samples

5.1.1.4 VAE-GAN based Synthetic Data Generation Model (Proposed Model)

The VAE-GAN architecture that is used for smart home synthetic data generation is shown in Figure. 5.1. This network architecture includes a GAN network with the generator module being a VAE neural network. As previously stated, the vanilla GAN network suffers from mode collapse. The main reason is that the discriminator is trapped in a local minimum, and the generator module repeatedly produces the output that is most likely to mislead the discriminator. As a result, training the GAN network becomes challenging and problematic. To address this problem, Larsen et al. [31] inserted a variational autoencoder into the GAN's generator module to leverage the VAE latent space's regularity.

The **encoder** module (E) compresses the input sequence into two vectors that are $mean_z$ and $variance_z$ of a Gaussian distribution by minimizing L_{prior}.

The **generator** module (G) reconstructs the input sequence from the latent space z so that the

reconstructed and original sequences have the lowest Mean Squared Error (MSE) by minimizing L_{VAE}. In addition, the input sequence cannot be considered as a generated sequence by the discriminator module, hence L_{dG} must be kept to a minimum. $L_{Generator}$ is computed as in Equation 5.7.

The **discriminator** module (D) needs to distinguish the original input sequence (L_{Real}) from the generator output sequence (L_{fake}). Meanwhile, to prevent the discriminator from failing to converge, L_{noise} is added to the discriminator's loss function. This term enforces D to distinguish a random sample from normal distribution from the real input sequence. The overall discriminator loss function is computed based on Equation 5.11.

$$L_{dG} = E_x[log(D(G(z)))] \tag{5.6}$$

$$L_{Generator} = L_{VAE} + L_{dG} \tag{5.7}$$

$$L_{real} = E_x[log(D(x))] \tag{5.8}$$

$$L_{fake} = E_z[1 - log(D(G(z)))] \tag{5.9}$$

$$L_{noise} = E_z[1 - log(D(\mathcal{N}(0, 1)))] \tag{5.10}$$

$$L_D = L_{real} + L_{fake} + L_{noise} \tag{5.11}$$

Figure. 5.2 presents the encoder structure of VAE-GAN. The generator and discriminator modules have similar structures, as shown in Figure. 5.3. Dilated one-dimensional convolutional (**Dilated**

CONV1D) neural network is used in the structure of the encoder, generator, and discriminator of the VAE-GAN network. This architecture is inspired by the WaveNet network [46] and utilizes dilated causal convolution layers to capture long-term dependencies in the input sequence. The 1-dimensional convolution slides a filter on an input series by one stride. However, in the dilated convolution, the sliding filter skips the input sequence with certain steps while keeping the order of the input data. Furthermore, multiple stacked dilated convolutional layers allow for longer input sequences, which reduces network complexity and training time compared with other long-term learning neural networks.

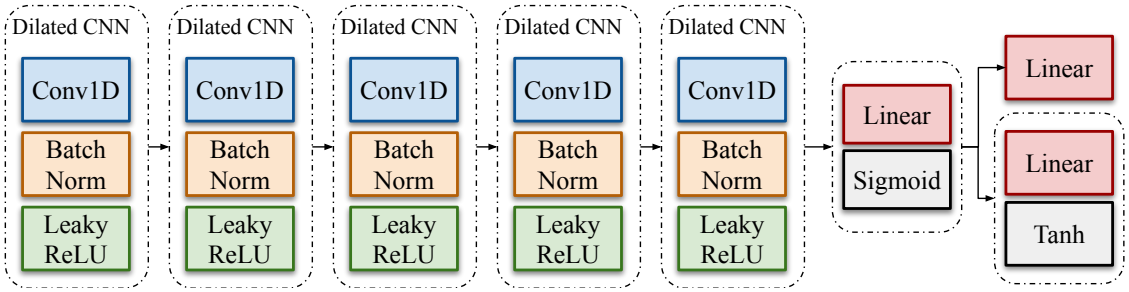


Figure 5.2: VAE-GAN encoder module structure

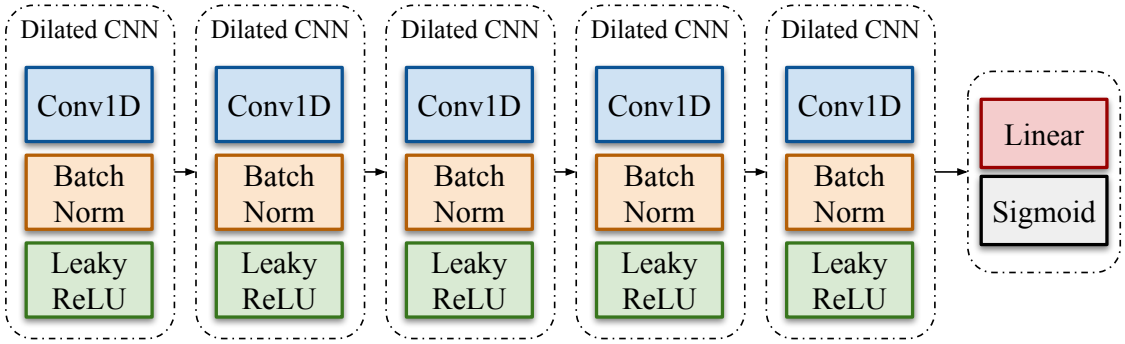


Figure 5.3: VAE-GAN generator and discriminator modules structure.

5.1.2 Evaluation Study

Here we will discuss the setup of our experiment, the dataset we used to evaluate the performance of our synthetic data generative model, the baseline models against which we measured the performance of our model, and finally our evaluation results and discussion of the results.

5.1.2.1 Experiment Setup

We use a real-world dataset, the iHomeLab RAPT dataset, to conduct our research [22]. This dataset includes residential electrical consumption data in appliance-level and aggregated household-level and solar panel (PV) energy production for five households in Switzerland spanning a period of 1.5 to 3.5 years with 5 minutes sampling frequency.

The residential house we selected from the dataset has 594 days worth of training after data cleansing. We use aggregated energy consumption and PV power production data with a 15-minute resolution in our training process. A household's historical energy consumption and PV generation data are both time-series data. We feed the data to the network in a sequence of 96 consecutive points to preserve the data characteristics in temporal order (a whole day).

In addition to comparing our method with the real data distributions, we used the vanilla GAN neural network, which is another well-known deep learning-based generative model, as a baseline to evaluate the performance of the proposed synthetic smart home data generator model.

5.1.2.2 Performance Measures

In this section, we introduce several metrics to evaluate the data generation performance of our technique.

Kullback–Leibler (KL) divergence

The KL divergence metric is used to determine the matching distance between two probability distributions. Equation 5.2 shows the definition of KL divergence, where $p(x_i)$ and $q(y_i)$ are probability distributions. $D_{KL}(p\|q) = 0$ represents two perfectly matched probability distributions that have identical quantities of information, and $D_{KL}(p\|q) = 1$ represents two completely different probability distributions.

Maximum Mean Discrepancy (MMD)

The MMD metric applies a kernel to determine the distances between two distributions based on the similarity of their moments. We use the radial basis function (RBF) defined as Equation 5.13 as the kernel in our experiments. Given the distributions $p(x)$ for $\{x_i\}_{i=0}^{N}$ and $q(y)$ for $\{y_j\}_{j=0}^{M}$, the MMD measure is calculated according to:

$$MMD(p,q)^2 = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}K(x_i,x_j) - \frac{2}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M}K(x_i,y_j) + \frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M}K(y_i,y_j) \quad (5.12)$$

$$K(x,y) = \exp(\frac{-\|x-y\|^2}{2\sigma^2}) \quad (5.13)$$

Wasserstein Distance

The Wasserstein distance between the distributions $p(x)$, $q(y)$ is computed by:

$$l_1(p,q) = \inf_{\pi\in\Gamma(p,q)}\int_{\mathbb{R}\times\mathbb{R}}|x-y|d\pi(x,y) \quad (5.14)$$

Here $\Gamma(p, q)$ is the set of all pairs of random variables $\pi(x, y)$ with respective cumulative distributions p and q. This metric computes the amount of distribution weight multiplied by the distance it has to be moved to transform distribution p to q.

Statistical parameters

[69] quantifies the load shape of a building's daily electricity consumption using five essential parameters:

- Near-peak load: p_{peak} is any load value that exceeds 97.5 percent of the load measurements.
- Near-base load: p_{base} is the $2.5th$ percentile of daily load.
- High-load duration: duration of having constant load with values close to the near-peak load.
- Rise time: the amount of time needed from reaching near-peak load from near-base.
- Fall time: the amount of time needed to fall from near-peak load to near-base load.

We compute the mean and standard deviation of these essential parameters for each generative model along with the real-world electrical load and PV production data for better comparison.

5.1.2.3 Evaluation Results and Discussion

Table 5.1 summarizes the KL divergence, Wasserstein distance, and MMD results between the real-world data and the synthetic data generated by GAN, and VAE-GAN networks.

The synthetic data generated by the VAE-GAN network (PV production and load consumption) have almost the same distribution as the real data, according to KL divergence results. As the

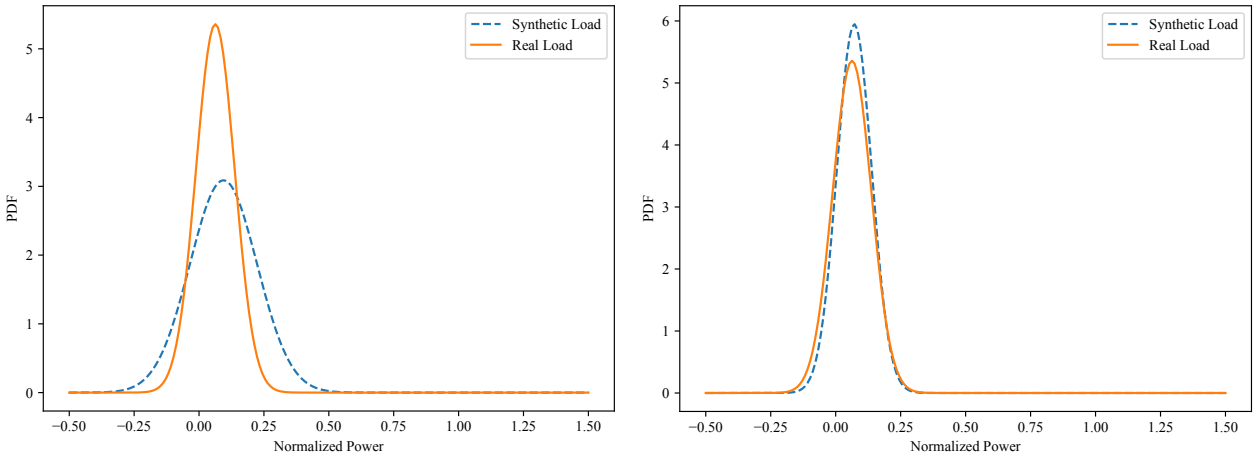
KL divergence inclines more toward zero, the two probability distributions are more similar. The probability density functions illustrated in Figure. 5.4 for electrical load consumption synthetic and real data, and Figure. 5.5 for PV power production synthetic and real data, support the same claim. This demonstrates that the VAE-GAN network is capable of learning the smart home data distribution and producing plausible samples that reflect the same distribution. PV production data has a lower KL divergence than load consumption data for both networks, which is expected given that PV production data follows the sunrise and sunset pattern. It is still highly affected by environmental factors but has more seasonality than electrical load consumption data.

Table 5.1: Distance between real and synthetic smart grid data distribution provided by vanilla GAN and VAE-GAN generative models.

Model	KL divergence		Wasserstein distance		MMD	
	Load	PV	Load	PV	Load	PV
GAN	0.543	0.273	530.3	961.5	0.227	0.224
VAE-GAN	0.017	0.006	255.3	266.6	0.101	0.117

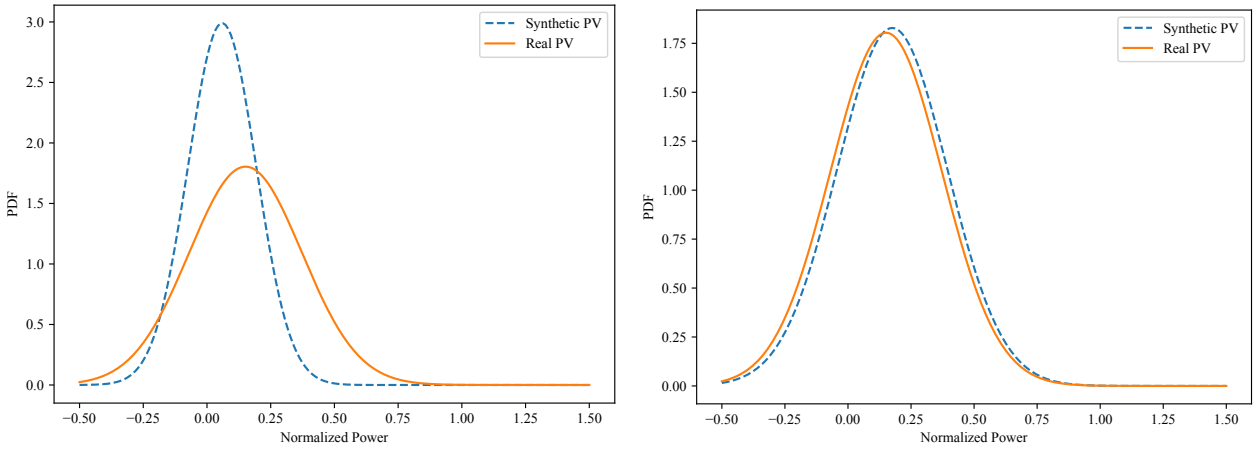
Wasserstein distance between synthetic data generated by the VAE-GAN network and real-world data is about 52% and 72% less compared to the data generated by the GAN network for electrical load consumption and PV production, respectively. This metric reveals that, compared with the GAN network, the distribution of synthetic data generated by the VAE-GAN network is considerably closer to the true distribution. Wasserstein distance values are large since the smart home data spectrum is between 0 and 15 kW.

MMD results, in Table 5.1, show the differences between synthetic data and real data distributions are lower for the VAE-GAN network, demonstrating the network's superior performance in generating synthetic data distributions compared to the GAN network.



(a) GAN (b) VAE-GAN

Figure 5.4: Electrical load consumption real and synthetic data probability density function for (a) GAN, and (b) VAE-GAN generative models. The orange line shows the real data PDF, the blue line shows the synthetic data PDF.



(a) GAN (b) VAE-GAN

Figure 5.5: PV power production real and synthetic data probability density function for (a) GAN, and (b) VAE-GAN generative models. The orange line shows the real data PDF, the blue line shows the synthetic data PDF.

Table 5.2 and 5.3 summarize the mean and standard deviation of five essential statistical parameters for synthetic and real electricity consumption and PV production data, respectively. For each parameter, we bold the result that is closer to the true distribution, for convenience. The data is min-max normalized before the training phase, then the base load is extremely close to zero. This is because of the difference in base load mean and standard deviation for synthetic and actual data for aggregated load consumption. The VAE-GAN network's ability to learn the load consumption and PV production data pattern are demonstrated by better peak load and high load duration results. Both models perform similarly and satisfactorily for the rise and fall time parameters.

Table 5.2: Overall load consumption statistical parameters evaluation results for synthetic data provided by GAN and VAE-GAN generative models versus the real data. The numbers that are closest to the real data are highlighted in bold.

Model	Base Load		Peak Load		High-Load Duration		Rise Time		Fall Time	
	mean	std	mean	std	mean	std	mean	std	mean	std
GAN	**3.53**	**22.34**	257.84	1658.12	**0.01**	0.08	**0.45**	**0.80**	**0.49**	**0.96**
VAE-GAN	1.73	10.99	**119.85**	**752.92**	**0.00**	0.04	**0.44**	0.74	0.52	**0.97**
Real data	9.75	51.13	151.39	1008.20	0.02	0.33	0.48	0.87	0.49	0.91

Table 5.3: PV power production statistical parameters evaluation results for synthetic data provided by GAN and VAE-GAN generative models versus the real data. The numbers that are closest to the real data are highlighted in bold.

Model	Base Load		Peak Load		High-Load Duration		Rise Time		Fall Time	
	mean	std	mean	std	mean	std	mean	std	mean	std
GAN	**0.00**	**0.00**	148.53	927.53	**0.02**	**0.50**	**0.49**	0.91	0.45	0.81
VAE-GAN	**0.04**	0.29	**205.63**	**1283.73**	**0.00**	0.00	0.42	0.79	**0.54**	**1.02**
Real data	0.00	0.00	197.91	1236.58	0.01	0.35	0.73	5.46	0.68	4.80

5.2 Q-Learning based Control Scheme with VAE-GAN based Synthetic Data Generation Model

The overall architecture of the proposed approach is described in Fig. 5.6. Given a real-world smart home dataset, we first apply the VAE-GAN-based method to produce synthetic data, including the smart home load, PV generation, and EV charging load consumption. Next, the generated data is used for the offline training of the Q-learning-based HEMS agent. Such an intelligent agent interacts with a synthetic data-based environment and produces HEMS strategies to maximize the long-term profit. Finally, the trained HEMS agent will be tested in an environment based on real-world data. We use the results to further investigate how the synthetic data quality will affect the HEMS performance.

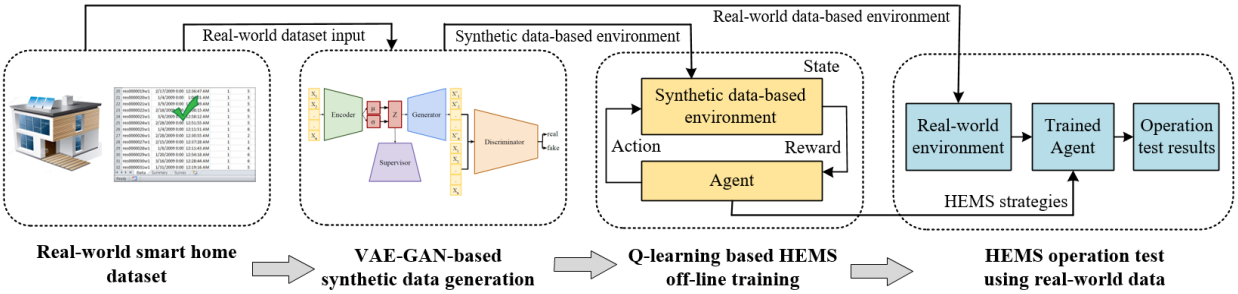


Figure 5.6: Overall architecture of the proposed Q-Learning based control scheme with VAE-GAN based synthetic data generation model.

5.2.1 Improved VAE-GAN based Synthetic Data Generation Model

To generate synthetic smart home data we adopted a VAE-GAN architecture, as presented in Figure. 5.7 that avoids the shortcomings of the vanilla GAN or VAE. For example, vanilla GAN is prone to mode collapse, a situation where the generator finds an input sequence that fools the discriminator and repeatedly produces it over and over again. The VAE-GAN architecture, introduced by Larsen et al. [31], generates new data samples based on a regulated latent space rather

than producing new data samples from a noise input. The discriminator will further classify the generated sample. Considering that the data for smart homes is time-series data, it is essential that the encoder be able to not only detect the underlying pattern and distribution in the input data but also retain the realistic order in which events occur. As a result, we took TimeGAN's [78] concept of incorporating a supervisor module into our VAE-GAN architecture.

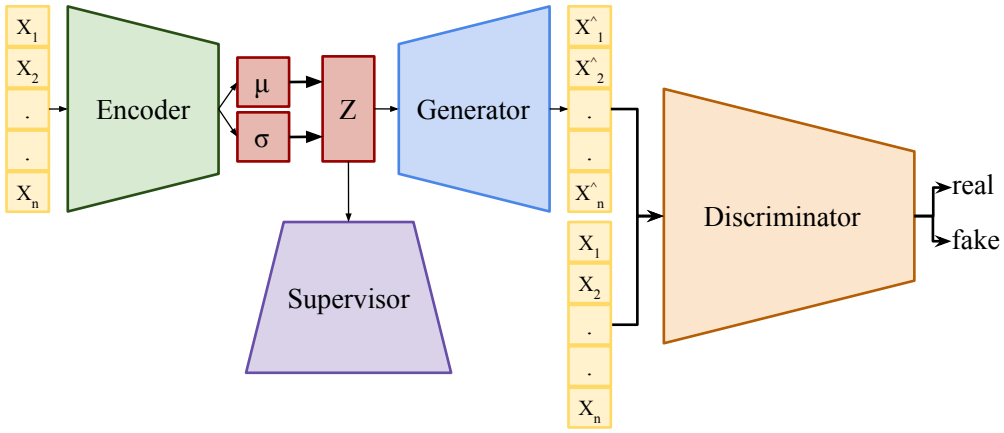


Figure 5.7: Improved VAE-GAN model architecture. In this network, the encoder module encodes the input sequence as a Gaussian distribution over the latent space, defined by mean and variance vectors. The supervisor module, trains the encoder module to approximate the next time step closely in the latent space. In the generator module, the input sequence is reconstructed from the latent space in an attempt to fool the discriminator so the generated sequence is considered real. The discriminator module trains the generator module to create realistic sequences by identifying fake samples from real ones.

We will discuss each module in more detail:

The **encoder** module (E) translates the original input sequence x into two vectors that represent the mean (μ) and variance (σ) of a standard Gaussian distribution. Minimizing $\mathcal{L}_{prior}$ (Equation 5.3) forces the encoder to compress the data over a standard Gaussian distribution. μ and σ are mapped into the latent space z using the *reparameterization* technique. The encoder uses five layers of **Dilated One Dimensional Convolutions** (dilated Conv1D) stacked one on top of the other (see

Figure. 5.8a). This architecture promises a similar performance but lower complexity and faster convergence compared to the Long Short-Term Memory (LSTM) networks typically used for time-series analysis. The Conv1D layer architecture resembles the WaveNet architecture [46], as each layer has a stride length of one and kernel size of two, but the dilation rate varies in each layer. As the second layer dilates at any second of the input sequence (skipping every other timestep), the third layer dilates at any fourth, etc., the lower layers are more focused on short-term dependencies while the higher layers capture long-term dependencies.

The **supervisor** module computes the distance between the latent representation at the current time step (z) and the next time step ($\hat{z}$), minimizing $\mathcal{L}_{supervisor}$. Added to the encoder loss function, Equation 5.16, allows the encoder to approximate the next time step in the latent space, retaining the original sequence of events.

$$\mathcal{L}_{supervisor} = \|z - \hat{z}\|^2 \tag{5.15}$$

$$\mathcal{L}_E = \mathcal{L}_{prior} + \mathcal{L}_{supervisor} \tag{5.16}$$

The **generator** module (G) is trained to reconstruct the original input sequence, given latent space z as the input by minimizing the Mean Squared Error (MSE) between the reconstructed sequence and original input sequence ($\mathcal{L}_{reconstr} + \mathcal{L}_{prior}$). The generator loss also contains the term $\mathcal{L}_{dG}$, computed as in Equation 5.18, quantifying the likelihood of the discriminator classifying the reconstructed sequence as a fake sequence. The goal is to make the reconstructed sequence so realistic that it can mislead the discriminator.

The **discriminator** module (D) is responsible for classifying the original input sequence and fake data samples. D minimizes $\mathcal{L}_D$, Equation 5.22, in the training process, which has three terms. $\mathcal{L}_{real}$, Equation 5.19, is the likelihood of original input data being classified as fake data, $\mathcal{L}_{fake}$, Equation 5.20, is the likelihood of reconstructed data sample being classified as real, and $\mathcal{L}_{noise}$,

Equation 5.21, is the likelihood of classifying a random noise input as a real data which is added to $\mathcal{L}_D$ to improve the convergence of the discriminator.

$$\mathcal{L}_{dG} = E_x[log(D(G(z)))] \tag{5.17}$$

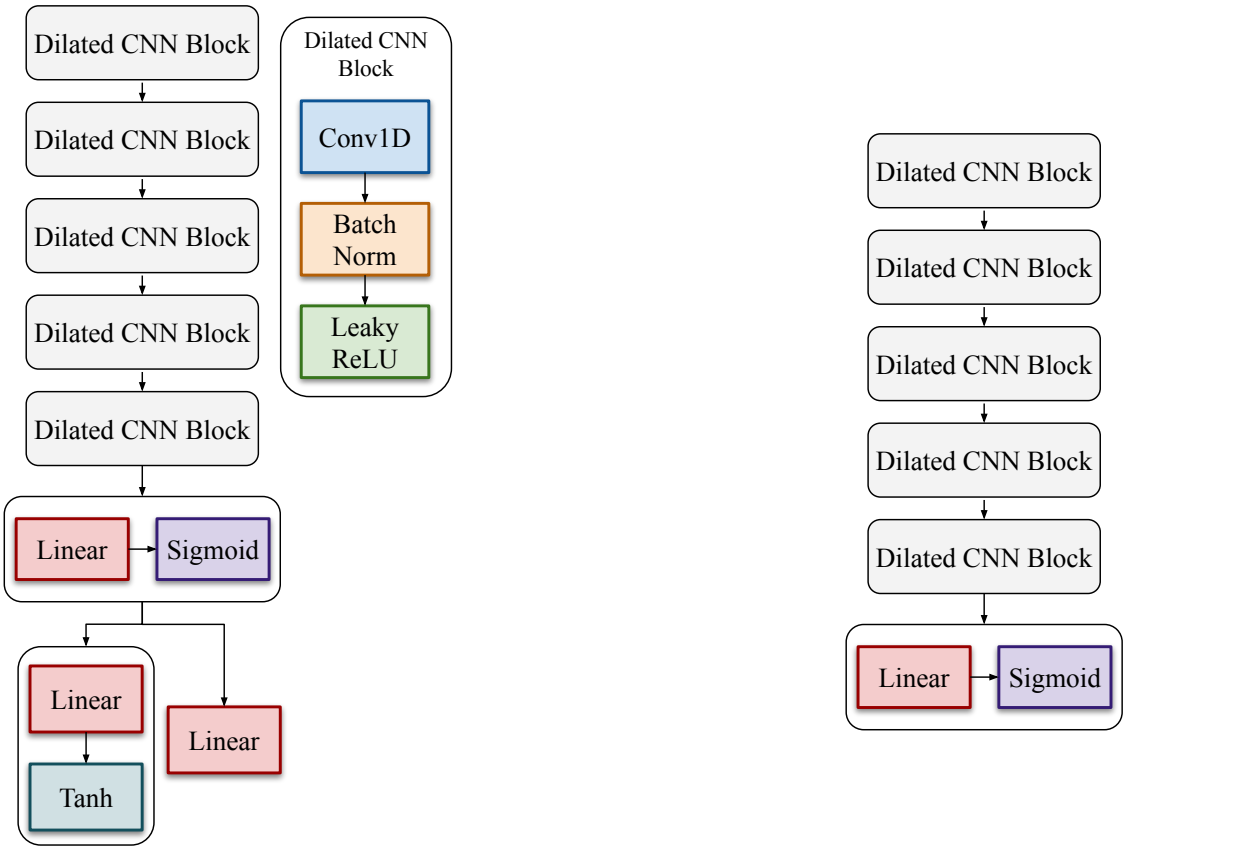
$$\mathcal{L}_{Generator} = \mathcal{L}_{prior} + \mathcal{L}_{reconstr} + \mathcal{L}_{dG} + \mathcal{L}_{supervisor} \tag{5.18}$$

$$\mathcal{L}_{real} = E_x[log(D(x))] \tag{5.19}$$

$$\mathcal{L}_{fake} = E_z[1 - log(D(G(z)))] \tag{5.20}$$

$$\mathcal{L}_{noise} = E_z[1 - log(D(\mathcal{N}(0,1)))] \tag{5.21}$$

$$\mathcal{L}_D = \mathcal{L}_{real} + \mathcal{L}_{fake} + \mathcal{L}_{noise} \tag{5.22}$$



(a) Encoder module structure

(b) Generator, supervisor, discriminator modules structure

Figure 5.8: Improved VAE-GAN modules structure

5.2.2 Q-learning based HEMS control

In this section, we introduce the Q-learning-based HEMS model. As shown in Figure. 5.9, the components of the smart home involved in the energy production and consumption include the PV panels, loads, EV, and the ESS:

- **PV**: The PV is considered a pure energy supplier. The PV power will first serve the energy demand of EV and other smart home devices, then the surplus energy can be used for ESS charging or selling to the energy trading market for profit.

- **EV and other smart home loads**: EV and other smart home load are pure energy consumers. They will first receive the power supply from PV or ESS for operation or buy electricity from the market if the internal power supply is insufficient.

- **ESS**: Finally, the ESS can be a power supplier when discharging, or a consumer when charging. For charging, it uses the surplus PV power or buys electricity from the market. For discharging, it supplies energy to EV and smart home devices or sells electricity to the energy trading market for profit.

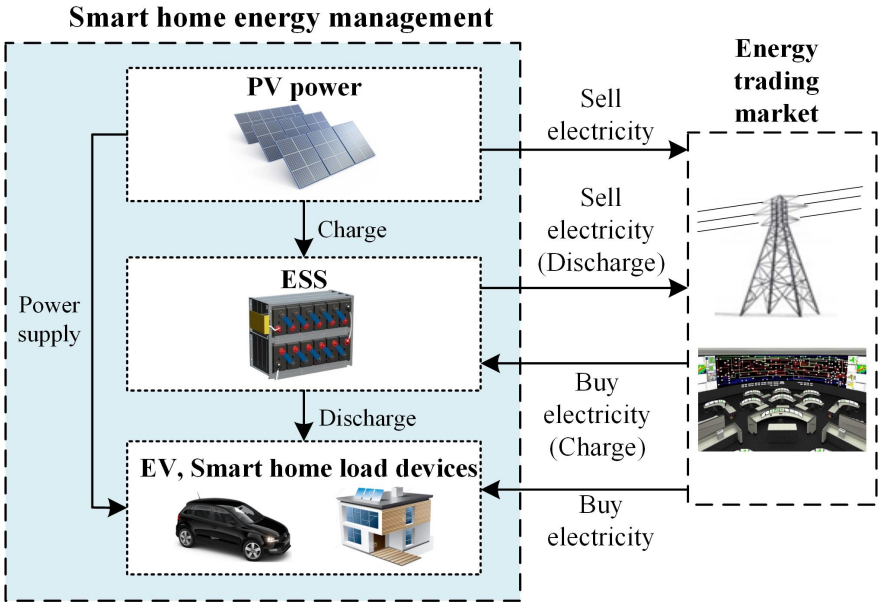


Figure 5.9: The proposed smart home energy management system

The HEMS model takes advantage of the flexibility of ESS to minimize the total energy cost and maximize profit. The optimization task of the centralized HEMS model is described as follows:

$$\max \quad \sum_{t=1}^{T} P^{total}(\gamma p_t^{sell} + (1-\gamma) p_t^{buy}) \tag{5.23}$$

$$\text{s.t.} \quad P^{total} = P_t^{ESS} + P_t^{PV} - P_t^{L} - P_t^{EV} \tag{5.23a}$$

$$\gamma = \mathbb{1}\{P^{total} > 0\} \tag{5.23b}$$

$$P_t^{ESS} = P_{ch} q_t \tag{5.23c}$$

$$q_t = \begin{cases} -1 & ESS \quad charges \\ 0 & ESS \quad unchanged \\ 1 & ESS \quad discharges \end{cases} \tag{5.23d}$$

$$Soc_{t+1} = Soc_t - \frac{P_t^{ESS}}{C^{ESS}} \tag{5.23e}$$

$$Soc_{min} \leq Soc_t \leq Soc_{max} \tag{5.23f}$$

where T is the total optimization period, p_t^{sell} and p_t^{buy} represent the price of selling energy to energy trading market and buying energy from the market, respectively. P_t^{ESS}, P_t^{PV}, P_t^{L} and P_t^{EV} represent the power of ESS, PV, smart home load, and EV charging load at time slot t, respectively. P^{total} denotes the total power consumption/demand of the smart home, and (5.23a) is the energy balance constraint. $\mathbb{1}$ is an indicator function. $\mathbb{1}\{P^{total} > 0\} = 1$ when $P^{total} > 0$, which means the agent will sell surplus energy for profit, otherwise $\mathbb{1}\{P^{total} > 0\} = 0$ if $P^{total} < 0$, which means buying energy from the market. P_{ch} is the fixed ESS charging power. $q_t = -1, 0, 1$ when ESS charges, remain unchanged and discharges, respectively. C^{ESS} is the fixed capacity of ESS, and Soc is the ESS state of charge (SOC). Equation 5.23c to 5.23e are ESS operation constraints, and Equation 5.23f is the SOC upper and lower bound constraint.

To transform this optimization task to the context of Q-learning, we define a Markov decision process (MDP) as follows:

- **State:** The agent state is defined as $s_t = \{Soc_t, P_t^{PV}, P_t^{Load}\}$, where $P_t^{Load} = P_t^L + P_t^{EV}$ represents the total energy consumption of smart home.

- **Action:** Based on the total energy demand and PV power generation, the agent decides the action $a_t = q_t$, which indicates the charging, discharging, or remaining unchanged status of ESS.

- **Reward:** By selecting actions intelligently, the agent intends to maximize the total profit in the optimization period, and the reward function is defined by:

$$r_t = P^{total}(\gamma p_t^{sell} + (1-\gamma)p_t^{buy}), \tag{5.24}$$

  which is the objective function of our problem formulation Equation 5.23.

Finally, the Q-learning-based HEMS algorithm is summarized in Algorithm 2, which consists of two phases. In the offline training phase, given the generated synthetic data, the intelligent agent is trained to maximize the total profit. Then, in the online test phase, we apply real-world data for online operation, which aims to test the performance of agents that are trained on synthetic data from various datasets.

Algorithm 2 Q-learning for HEMS

Initialize: Q-learning and smart home parameters
Phase 1: Off-line training using synthetic data:
  Input: generated synthetic data of smart home load, EV load and PV power generation.
  for $episode = 1$ to E **do**
    With probability ϵ choose action a randomly. Otherwise, $a = arg\ max(Q(s, a))$
    Agent calculates reward based on equation (4.9).
    Update agent state $\{t, S_t, w_{g,t}\}$ and Q-value:
    $Q(s, a) = (1 - \alpha)Q(s, a) + \alpha(r + \gamma maxQ(s', a'))$
  end for
Output: Trained Q-learning based HEMS strategy.
Phase 2: Online test using real-world data:
  Input: Real-world data.
  Deploying the pre-trained Q-learning agent for operation test.
Output: HEMS profit under real-world data.

5.2.3 Evaluation Study

We will discuss the settings of our experiment, the dataset we used to train the synthetic data generative model and evaluate our Q-learning based HEMS, the baseline models against which we measured the performance of our models, and finally our evaluation results and discussion of our findings.

5.2.3.1 Experiment Setup

SmartHome Dataset

The iHomeLab RAPT dataset [22] is a real-world dataset for residential power traces. Five households in Switzerland have been surveyed for 1.5 to 3.5 years with a sampling frequency of 5 minutes. The dataset contains residential electricity consumption and PV energy production, including appliance-level and aggregated household consumption data. Studies of the PV power generation show recurring patterns with corresponding noises originating from rain and clouds. Consumption patterns show weekly trends, but with occasional irregular spikes.

The experiment was conducted on the residential house D from the dataset. Data cleansing was done by choosing days with full 24-hour records, resulting in 594 days in total. As part of the training process, electrical load and PV power production data were downsampled to a resolution of 15 minutes.

Residential Electric Vehicle Charging Dataset

Sorensen et al. [57] presented a residential electric vehicle charging dataset recorded from apartment buildings. From December 2018 to January 2020, 97 users in Norway reported real-world EV charging experiences. Each charging session includes plug-in time, plug-out time, and charged energy. This dataset provides a synthetic charging load for level-1 charging and level-2 charging assuming 3.6 kW or 7.2 kW of charging power. In our experiments, we interpolate the data for a 15-minute resolution.

Our analysis revealed that the user with the largest number of available records has only 62 days of 24-hour data. This data contains only 11 charging sessions, which is insufficient for the training of a deep learning model. To address this issue we generated a larger synthetic dataset by combining the charging data from several users and assuming that the data belonged to an indoor charging station shared by the building's residents.

Baseline Models

In our experiments, we compared the performance of our improved VAE-GAN-based approach with two other state-of-the-art approaches:

- **Gaussian Mixture Model (GMM)**: is a probabilistic approach that partitions data into groups using soft clustering based on multiple multidimensional Gaussian probability distributions. The mean and variance of the distributions are calculated using Expectation-Maximization. Upon fitting the GMM to some data, a generative probabilistic model can sample synthetic data, following the same distribution.
- **Vanilla GAN**: A well-known deep learning-based generative model, as described in Sec-

tion 5.1.1.3.

- **Improved VAE-GAN**: As described in Section 5.2.1

5.2.3.2 Performance Measures

In the following, we introduce the metrics that we will employ to evaluate the quality of the generated synthetic data.

Kullback–Leibler (KL) divergence

The Kullback-Leibler divergence (KLD) is one of the most often used measures for evaluating the similarity of two probability distributions. Equation 5.25 is the formal definition of KL divergence, where x and y are sampled data points, and $p(x)$ and $p(y)$ are their respective probability distributions. The KL divergence ranges from 0 when two probability distributions almost match everywhere, to ∞ for completely different distributions.

$$D_{KL}(p\|q) = \sum_{i=1}^{N} p(x_i) log\left(\frac{p(x_i)}{q(y_i)}\right) \tag{5.25}$$

Maximum Mean Discrepancy (MMD)

The MMD approach represents the distance between distributions by the distance between the mean embeddings of features into a reproducing kernel Hilbert space. Given the distributions $p(x)$ for $\{x_i\}_{i=0}^{N}$ and $q(y)$ for $\{y_j\}_{j=0}^{M}$, MMD is calculated as follows:

$$MMD(p,q)^2 = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} K(x_i,x_j) - \frac{2}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M} K(x_i,y_j) + \frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M} K(y_i,y_j) \tag{5.26}$$

$$K(x, y) = \exp\left(\frac{-\|x - y\|^2}{2\sigma^2}\right) \tag{5.27}$$

Wasserstein Distance

Intuitively, the Wasserstein distance also called the earth mover's distance, models a probability distribution as a pile of soil, and computes how much soil needs to be moved to transform one probability distribution to the other.

$$l_1(p, q) = \underset{\pi \in \Gamma(p,q)}{inf} \int_{\mathbb{R}\times\mathbb{R}} |x - y| d\pi(x, y) \tag{5.28}$$

Equation 5.28 represents the formal definition of Wasserstein distance, where $p(x)$, $q(y)$ are probability distributions, $\Gamma(p, q)$ is the set of probabilistic distributions on $\mathbb{R} \times \mathbb{R}$ whose marginals represent p and q on the first and second moments, respectively.

HEMS Models

The previous techniques measured the quality of the generated data based on its distance from the probability distribution of the real data. In this technique, we measure the quality of the synthetic data by its ability to be used in the training of a Q-learning-based smart home energy management system (HEMS). We implemented the Q-learning model in the MATLAB platform. The ESS fixed charging/discharging power is 4 kW, and the ESS capacity is 16 $kW \cdot h$. The learning rate is 0.8, the discount factor is 0.7, and the initial ϵ value for the ϵ-greedy exploration algorithm is 0.05. The energy trading price follows a fixed pattern as in [87]. We recorded the average results over 10 randomized runs.

Table 5.4: Distance between real and synthetic smart grid data distribution using KL-divergence, Wasserstein distance and MMD. Best results highlighted in **bold**.

Model	KL divergence			Wasserstein distance			MMD		
	Load	PV	EV	Load	PV	EV	Load	PV	EV
GMM	0.277	1.067	0.127	619.3	1319.3	0.006	0.168	0.074	0.000047
GAN	0.104	0.454	0.104	437.42	1080.5	**0.002**	**0.010**	**0.049**	**0.000028**
VAE-GAN	**0.065**	**0.071**	**0.085**	**295.458**	**648.87**	**0.005**	**0.011**	0.186	0.000071

5.2.3.3 Distance Metrics Evaluation Results and Discussion

We used the KL-divergence, MMD, and Wasserstein distance metrics to measure how close is the distribution of synthetic data to the real data. Table 5.4 shows these metrics for smart home electrical load consumption, PV production, and EV charging load consumption synthetic data generated by the GMM, GAN, and improved VAE-GAN generative models. For all metrics, the lower the values are the better, corresponding to a closer match between the synthetic and real data.

The results allow us to draw several conclusions. First, we find that the distances are much smaller for the EV value than for PV and the load. This might be because the nightly EV charging is comparatively regular compared to the PV generation that is affected by the weather or the load which depends on many personal choices. Second, we find that for the KL-divergence and Wasserstein distance, our improved VAE-GAN-based approach provides almost always the best performance. For the MMD, in contrast, the best performance is provided by the GAN approach. This reversal in the relative order of the generator approaches for MMD is likely a result of the choice of the feature embedding. Understanding the exact mechanism of this difference requires future work.

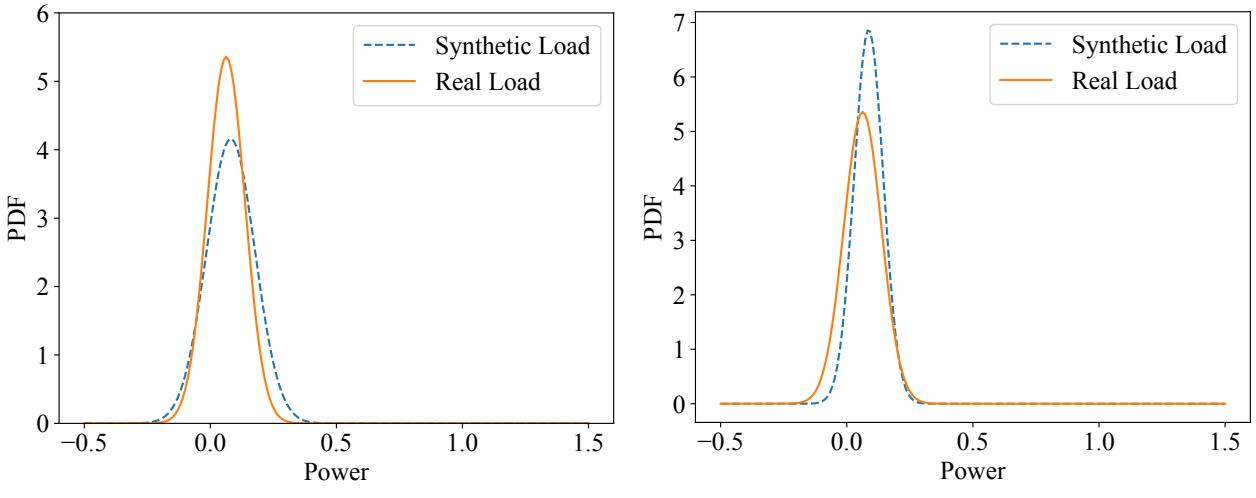
While single numerical metrics quantifying the distance between the probability density functions are useful, examining the distributions as a whole provides us with a better understanding of the

quality of the generated data. Figures 5.10 to 5.12 show the Probability Density Functions (PDF) for real and synthetic smart home data.

Figure. 5.10 shows the PDF for normalized real and synthetic electrical load consumption for each generative model. While both the GMM and VAE-GAN distributions are relatively close to the real data, the synthetic data generated by VAE-GAN provides the best matches with the real data distribution in terms of the standard deviation from the mean value, as supported by the distance metrics. Figure. 5.13a represents the pattern of real and synthetic electrical load consumption during the test data. It is evident from the close similarity of these patterns that the improved VAE-GAN network is capable of learning the distribution and patterns of smart home aggregated load consumption data and producing samples that reflect the same characteristics.

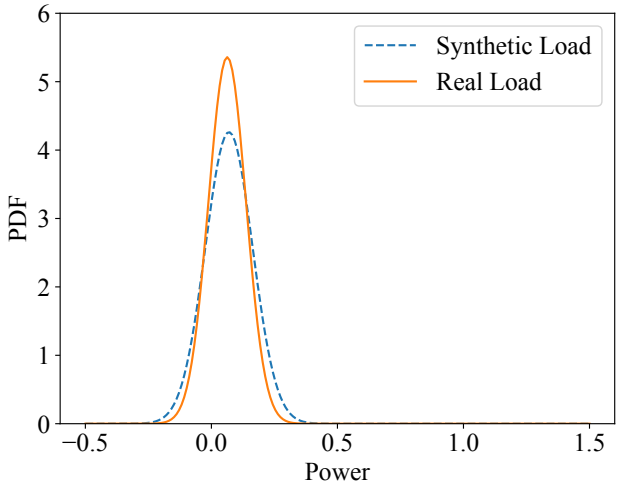
Figure. 5.11c illustrates that the improved VAE-GAN model performs better compared to two other generative models in generating synthetic PV production data, a conclusion that is supported by Figure. 5.13b, where the pattern of synthetic PV production data generated by improved VAE-GAN is reasonably close to the pattern of real PV production data. We note that although PV production follows sunrise and sunset patterns, it is also highly affected by unpredictable environmental factors.

Finally, Figure. 5.13c presents the EV charging load consumption real and synthetic data PDF for GMM, GAN, and improved VAE-GAN generative models. In Figure. 5.12a it is shown that synthetic EV charging load consumption generated by the GMM model is larger in power range compared to the real EV charging load consumption. On the other hand, the GAN model in Figure. 5.12b, generates data centered around the average. Although the improved improved VAE-GAN model, Figure. 5.12c, generates a slightly smaller power consumption than actual EV charging data, the distribution of load consumption is comparable to that of real data. Accordingly, each generative model shows the same characteristics in the sample EV data presented in Figure. 5.13c.



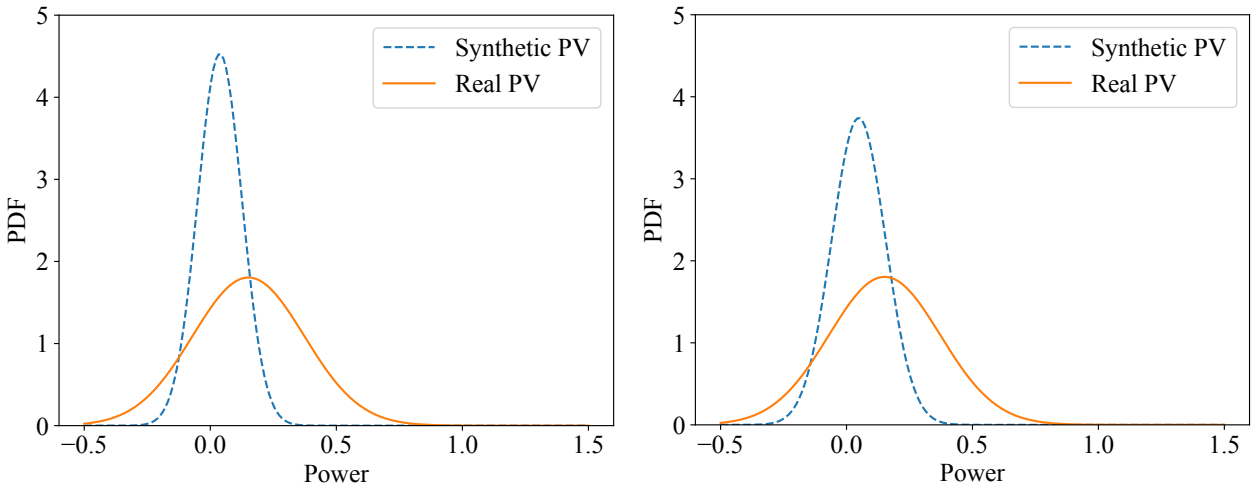
(a) GMM

(b) GAN



(c) VAE-GAN

Figure 5.10: Electrical load consumption real and synthetic data probability density for (a) GMM, (b) GAN, and (c) improved VAE-GAN generative models. The orange line shows the real data PDF, the blue line shows the synthetic data PDF.

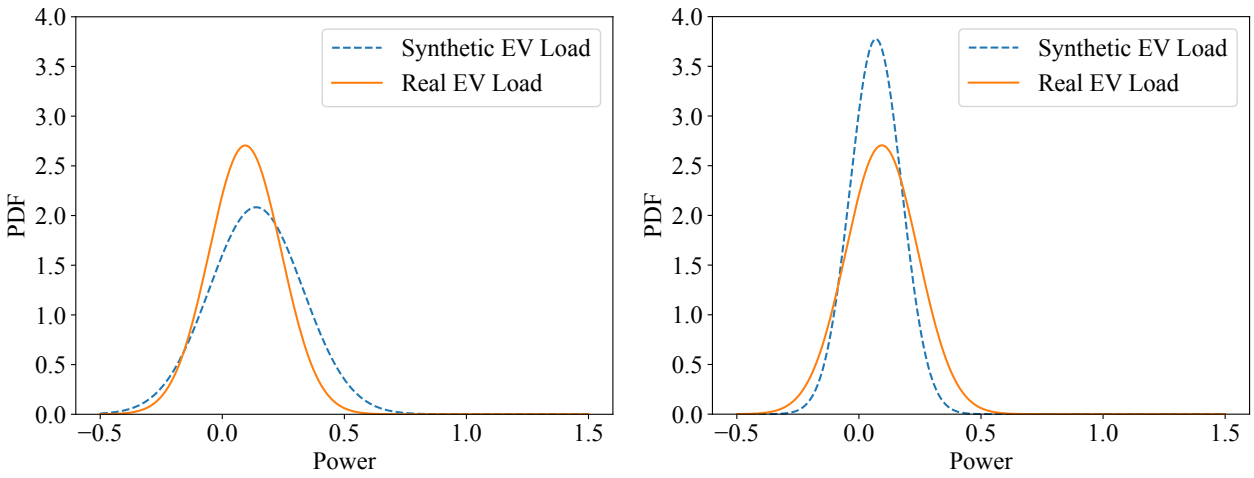


(a) GMM

(b) GAN

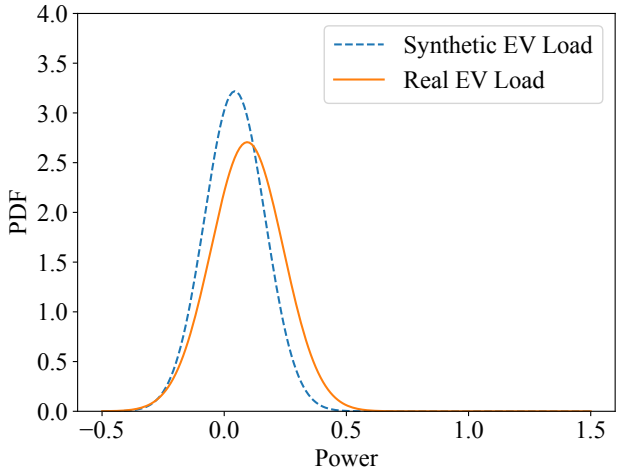
(c) VAE-GAN

Figure 5.11: PV power production real and synthetic data probability density for (a) GMM, (b) GAN, and (c) improved VAE-GAN generative models. The orange line shows the real data PDF, the blue line shows the synthetic data PDF.



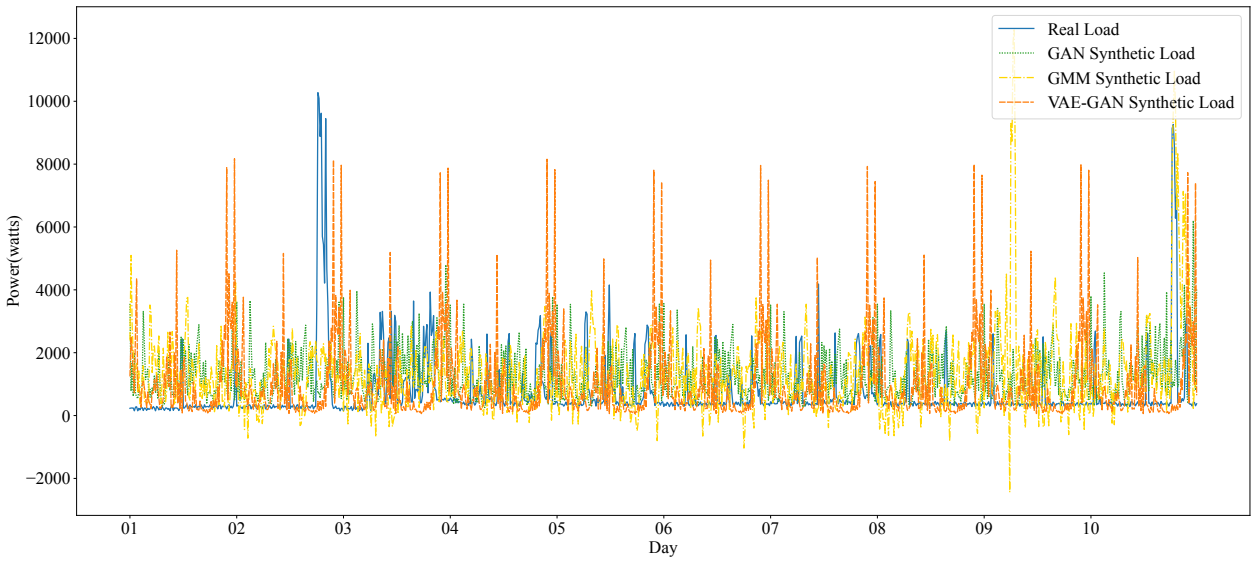
(a) GMM

(b) GAN

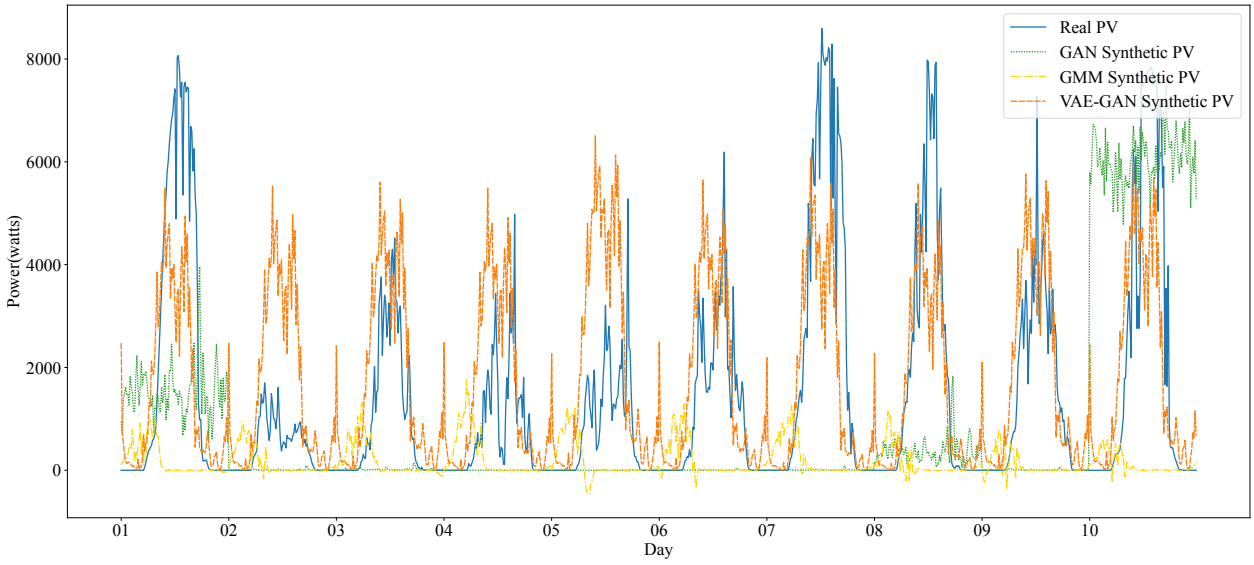


(c) VAE-GAN

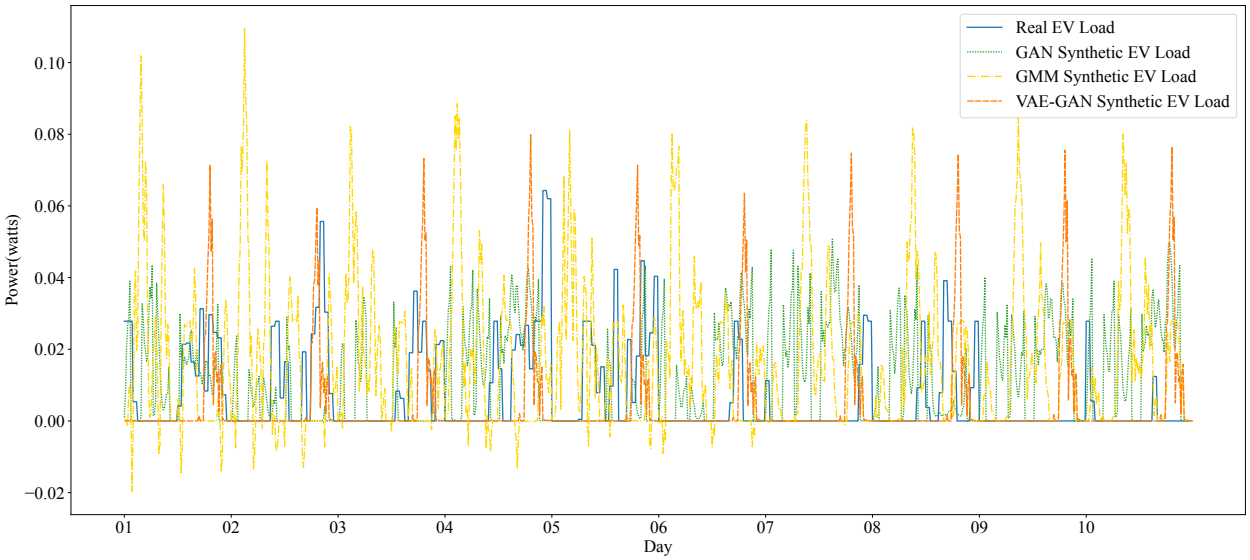
Figure 5.12: EV charging load consumption real and synthetic data probability density function for (a) GMM, (b) GAN, and (c) improved VAE-GAN generative models. The orange line shows the real data PDF, the blue line shows the synthetic data PDF.



(a) Aggregated Load



(b) PV Power Production



(c) EV Load Consumption

Figure 5.13: A sample of 10 days of synthetic data generated from GAN, GMM, and improved VAE-GAN models compared with the real test data.

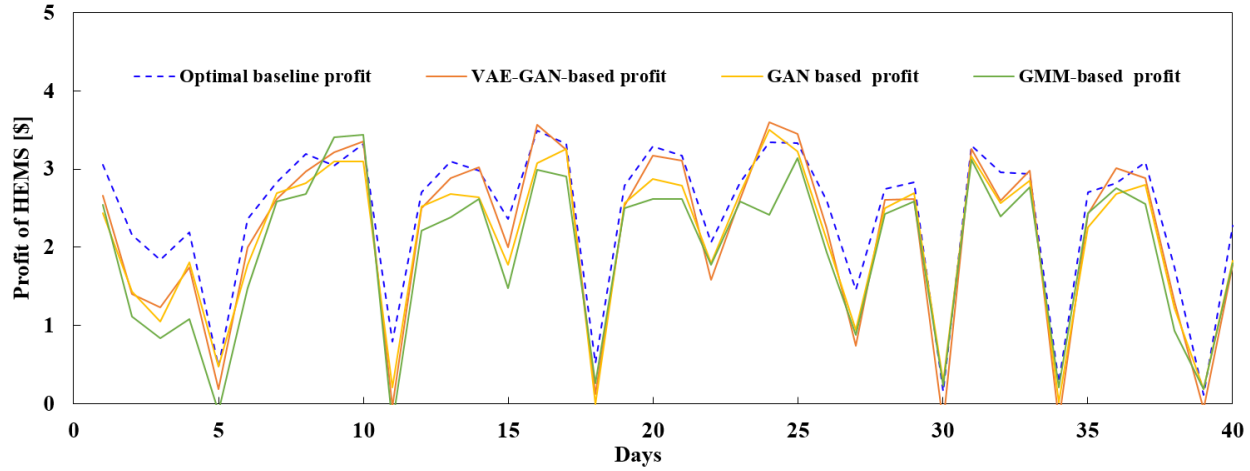


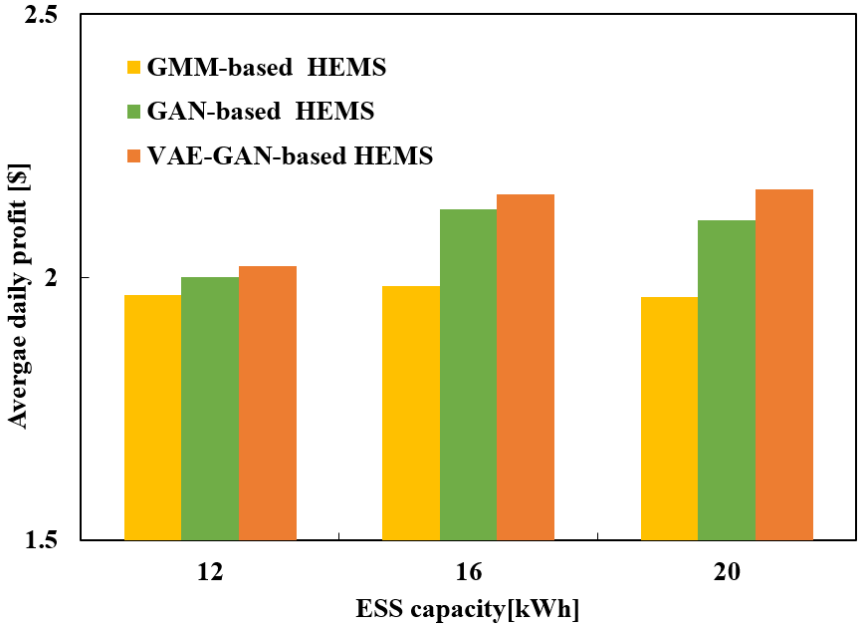
Figure 5.14: HEMS test performance comparison in 40 days by using real-world data (16 kWh ESS Capacity)

5.2.3.4 HEMS Performance Evaluation Results and Discussion

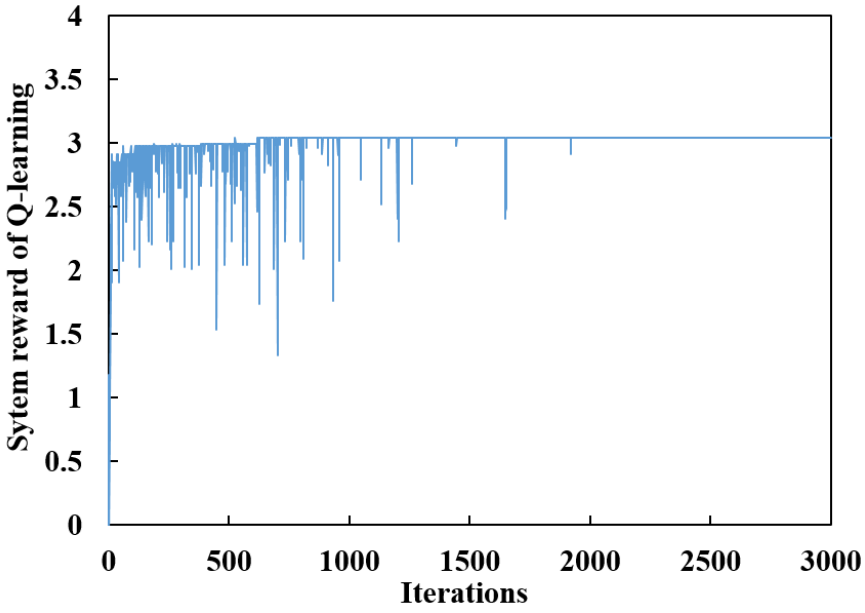
In this section, we investigate the performance of the Q-learning-based HEMS. For each type of synthetic data, we trained a corresponding policy, which then was evaluated using real-world data (see Algorithm 1). We compare these policies against an optimal baseline which is using the real-world data for both offline training and online operation testing.

Figure. 5.14 shows the 40 days HEMS profit of different data generation methods. One can observe that the proposed VAG-GAN-based HEMS achieves a comparable performance with the optimal baseline, which can be explained as the advantage of the improved VAE-GAN-based synthetic data generation scheme. Meanwhile, compared with improved VAE-GAN, GNN-based HEMS also presents comparable results. By contrast, the GMM-based method shows the lowest profit in 40 days, and the main reason lies in the low quality of generated synthetic data. The results in Figure. 5.14 demonstrate that the quality of generated synthetic data will affect the performance of HEMS, and our proposed improved VAE-GAN-enabled synthetic data generation algorithm can better contribute to a satisfying HEMS performance than GNN and GMM-based methods.

Meanwhile, we investigate the HEMS performance under various ESS capacities, which is illustrated in Figure. 5.15a. It shows that the proposed improved VAE-GAN-based HEMS outperforms GMM and GAN methods by higher average profit. On the other hand, increasing ESS capacity will bring more profit for improved VAE-GAN and GAN-based HEMS, since the agent has higher flexibility with higher ESS capacity. When the ESS capacity is 20 kWh, the improved VAE-GAN based HEMS achieves 10.4% and 2.8% higher profit than GMM and GAN-based methods, respectively. Moreover, the convergence performance of Q-learning based HMES is given in Figure. 5.15b. It demonstrates that the designed agent has a stable convergence performance after the exploration phase.



(a) Average daily HEMS profit comparison under various ESS capacities



(b) Convergence performance of Q-learning based HEMS in one episode

Figure 5.15: Smart home HEMS performance analyses

CHAPTER 6: CONCLUSION

The widespread adoption of home energy management systems (HEMS) has been boosted by advances in communication technology and information flow associated with home energy sources photovoltaic (PV) power generation, smart grid and plug-in electric vehicles. These systems are essential for improving energy efficiency and reducing energy consumption costs. A load forecasting algorithm assists systems such as these by providing a realistic prediction of future residents' energy needs so the system can schedule tasks that use the different energy sources more efficiently.

In chapter 3, we proposed an appliance-level load forecasting model for residential homes. This model uses LSTM recurrent neural networks to learn energy consumption patterns for individual electrical appliances in a smart home and predict a given appliance's potential load consumption over different time intervals. We evaluated our model on a public dataset compared with Random Forest and FFN network models. The RMSE and NRMSE results show that our model is more accurate than the baselines.

In chapter 4, we improved the appliance-level load forecasting model to an LSTM-based Sequence to Sequence learning model that forecasts the energy consumption of smart home appliances for the hour ahead with 10-minute resolution using historical data of the past day. We evaluated the performance of our model compared with VARMA, Dilated 1D Convolution, and LSTM on a publicly available dataset that contains historical data for multiple buildings. The LSTM-based sequence to sequence models consistently outperformed the other choices, demonstrating their wide applicability. Furthermore, we implemented Bi-LSTM networks in the encoder module of the Seq2Seq learning forecasting model to learn the underlying patterns among load data from both forward and backward timesteps. Moreover, we included the PV power generation as a new source of energy in our model and the outputs of the models are further fed into a Q-learning model for

offline HEMS optimization. The online performance of the optimization model is tested using the ground-truth values. The proposed model has outperformed other algorithms, with a narrow error margin between online and offline operations.

In chapter 5, we apply a variational autoencoder GAN (VAE-GAN) for synthetic time series generation of the smart home data, including electrical load and PV generation. The main advantage of this technique is that the network does not require any prior training analysis on the data, and can be utilized to generate different forms of smart home data, including household electricity load and PV power generation. The model performance and synthetic data effectiveness are assessed with respect to the vanilla GAN network. In the next step, we improved the accuracy of the VAE-GAN for maintaining the temporal order of energy consumption fluctuations. This would also include generating EV charging load consumption, as well as electrical load consumption and PV generation. Further evaluating our model, we considered its performance in conjunction with a proposed Q-learning-based smart home energy management system (HEMS). The VAE-GAN smart home synthetic data generative model is compared with a Gaussian Mixture Model and a vanilla GAN in both synthetic data generation and operation levels. The results of the experiments show that the VAE-GAN model outperforms other models in terms of distance metrics. With its tight error margin between online and offline operations, the suggested model outperforms the baseline algorithms.

LIST OF REFERENCES

[1] Abien Fred Agarap. “Deep learning using rectified linear units (relu)”. In: *arXiv preprint arXiv:1803.08375* (2018).

[2] Dima Alberg and Mark Last. “Short-term load forecasting in smart meters with sliding window-based ARIMA algorithms”. In: *Vietnam Journal of Computer Science* 5.3 (2018), pp. 241–249.

[3] Muhammad Saidu Aliero et al. “Smart Home Energy Management Systems in Internet of Things networks for green cities demands and services”. In: *Environmental Technology & Innovation* 22 (2021), p. 101443.

[4] Abdulaziz Almalaq and George Edwards. “A Review of Deep Learning Methods Applied on Load Forecasting”. In: *in Proc. of 2017 IEEE International Conference on Machine Learning and Applications*. Dec. 2017, pp. 1–6.

[5] Hamed HH Aly. “A proposed intelligent short-term load forecasting hybrid models of ANN, WNN and KF based on clustering techniques for smart grid”. In: *Electric Power Systems Research* 182 (2020), p. 106191.

[6] Kasun Amarasinghe, Daniel L. Marino, and Milos Manic. “Deep Neural Networks for Energy Load Forecasting”. In: *Proc. of 2017 IEEE International Symposium on Industrial Electronics (ISIE)*. June 2017, pp. 1–6.

[7] Kadir Amasyali and Nora El-Gohary. “A review of data-driven building energy consumption prediction studies”. In: *Renewable Sustainable. Energy Reviews* 81 (Jan. 2019), pp. 1192–1205.

[8] Altaf QH Badar and Amjad Anvari-Moghaddam. “Smart home energy management system–a review”. In: *Advances in Building Energy Research* 16.1 (2022), pp. 118–143.

[9] Hai-Hong Bian, Qian Wang, and Linlin Tian. “Research on Short-Term Load Forecasting Based on PCA-GM”. In: *International Conference on Multimedia Technology and Enhanced Learning*. Springer. 2020, pp. 169–177.

[10] Robert Buechler et al. “EVGen: Adversarial Networks for Learning Electric Vehicle Charging Loads and Hidden Representations”. In: *arXiv preprint arXiv:2108.03762* (2021).

[11] Siyun Chen et al. “Butler, not servant: A human-centric smart home energy management system”. In: *IEEE Communications Magazine* 55.2 (2017), pp. 27–33.

[12] Yize Chen et al. “Model-free renewable scenario generation using generative adversarial networks”. In: *IEEE Transactions on Power Systems* 33.3 (2018), pp. 3265–3275.

[13] Longquan Diao et al. “Modeling energy consumption in residential buildings: A bottom-up analysis based on occupant behavior pattern clustering and stochastic simulation”. In: *Energy and Buildings* 147 (2017), pp. 47–66.

[14] Pantelis Dimitroulis and Miltiadis Alamaniotis. “A fuzzy logic energy management system of on-grid electrical system for residential prosumers”. In: *Electric Power Systems Research* 202 (2022), p. 107621.

[15] Samer El Kababji and Pirathayini Srikantha. “A data-driven approach for generating synthetic load patterns and usage habits”. In: *IEEE Transactions on Smart Grid* 11.6 (2020), pp. 4984–4995.

[16] Mohammad Navid Fekri, Ananda Mohon Ghosh, and Katarina Grolinger. “Generating energy data for machine learning with recurrent generative adversarial networks”. In: *Energies* 13.1 (2019), p. 130.

[17] Cong Feng, Mucun Sun, and Jie Zhang. “Reinforced deterministic and probabilistic load forecasting via Q-learning dynamic model selection”. In: *IEEE Transactions on Smart Grid* 11.2 (2020), pp. 1377–1386.

[18] Shaoyun Ge et al. “Domestic energy consumption modeling per physical characteristics and behavioral factors”. In: *Energy Procedia* 158 (2019), pp. 2512–2517.

[19] Ian Goodfellow et al. “Generative adversarial nets”. In: *Advances in neural information processing systems* 27 (2014).

[20] Yunliang Hao, Wei Wang, and Yuli Qi. “Optimal home energy management with PV system in time of use tariff environment”. In: *2017 Chinese Automation Congress (CAC)*. IEEE. 2017, pp. 2693–2697.

[21] Ejaz Ul Haq et al. “Forecasting household electric appliances consumption and peak demand based on hybrid machine learning approach”. In: *Energy Reports* 6 (2020), pp. 1099–1105.

[22] Patrick Huber et al. “Residential power traces for five houses: the iHomeLab RAPT dataset”. In: *Data* 5.1 (2020), p. 17.

[23] Zhu Jian-an et al. “Short-term load forecasting model of gray-weighted markov chain based on particle swarm optimization”. In: *2020 Asia Energy and Electrical Engineering Symposium (AEEES)*. IEEE. 2020, pp. 825–830.

[24] Li Jiang, Da-You Liu, and Bo Yang. “Smart home research”. In: *Proceedings of 2004 international conference on machine learning and cybernetics (IEEE Cat. No. 04EX826)*. Vol. 2. IEEE. 2004, pp. 659–663.

[25] Michael J Kane et al. “Comparison of ARIMA and Random Forest time series models for prediction of avian influenza H5N1 outbreaks”. In: *BMC bioinformatics* 15.1 (2014), p. 276.

[26] Nikos Komninos, Eleni Philippou, and Andreas Pitsillides. “Survey in smart grid and smart home security: Issues, challenges and countermeasures”. In: *IEEE Communications Surveys & Tutorials* 16.4 (2014), pp. 1933–1954.

[27] Weicong Kong et al. “A practical solution for non-intrusive type II load monitoring based on deep learning and post-processing”. In: *IEEE Transactions on Smart Grid* 11.1 (2019), pp. 148–160.

[28] Weicong Kong et al. “Short-term residential load forecasting based on LSTM recurrent neural network”. In: *IEEE Transactions on Smart Grid* 10.1 (2017), pp. 841–851.

[29] Xiangyu Kong et al. “Improved Deep Belief Network for Short-Term Load Forecasting Considering Demand-Side Management”. In: *IEEE Transactions on Power Systems* 35.2 (Mar. 2019), pp. 1531–1538.

[30] Manu Lahariya, Dries F Benoit, and Chris Develder. “Synthetic data generator for electric vehicle charging sessions: Modeling and evaluation using real-world data”. In: *Energies* 13.16 (2020), p. 4211.

[31] Anders Boesen Lindbo Larsen et al. “Autoencoding beyond pixels using a learned similarity metric”. In: *Proc. of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. PMLR, 2016, pp. 1558–1566. URL: https://proceedings.mlr.press/v48/larsen16.html.

[32] Chen Li. “Designing a short-term load forecasting model in the urban smart grid system”. In: *Applied Energy* 266 (2020), p. 114850.

[33] Lechen Li et al. “Short-term apartment-level load forecasting using a modified neural network with selected auto-regressive features”. In: *Applied Energy* 287 (2021), p. 116509.

[34] Weixiang Li et al. “Implemented IoT-Based Self-Learning Home Management System (SHMS) for Singapore”. In: *IEEE Internet of Things Journal* 5 (June 2018), pp. 2212–2219.

[35] Andy Liaw, Matthew Wiener, et al. “Classification and regression by randomForest”. In: *R news* 2.3 (2002), pp. 18–22.

[36] Nguyen Viet Linh and Pablo Arboleya. “Deep Learning Application to Non-Intrusive Load Monitoring”. In: *Proc. of 2019 IEEE Milan PowerTech*. June 2019, pp. 1–6.

[37] Bhamidi Lokeshgupta and Shanmugavelu Sivasubramani. “Cooperative game theory approach for multi-objective home energy management with renewable energy integration”. In: *IET Smart Grid* 2.1 (2019), pp. 34–41.

[38] Juan Miguel Gonzalez Lopez et al. “Smart residential load simulator for energy management in smart grids”. In: *IEEE Transactions on Industrial Electronics* 66.2 (2018), pp. 1443–1452.

[39] Renzhi Lu, Seung Ho Hong, and Mengmeng Yu. “Demand response for home energy management using reinforcement learning and artificial neural network”. In: *IEEE Transactions on Smart Grid* 10.6 (2019), pp. 6629–6639.

[40] Mousa Marzband et al. “Optimal energy management system based on stochastic approach for a home Microgrid with integrated responsive load demand and energy storage”. In: *Sustainable cities and society* 28 (2017), pp. 256–264.

[41] Alwyn Mathew, Milan Jeetendra Jolly, and Jimson Mathew. “Improved residential energy management system using priority double deep Q-learning”. In: *Sustainable Cities and Society* 69 (2021), p. 102812.

[42] Fady Y Melhem et al. “Optimization and energy management in smart home considering photovoltaic, wind, and battery storage system with integration of electric vehicles”. In: *Canadian Journal of Electrical and Computer Engineering* 40.2 (2017), pp. 128–138.

[43] Andrea Monacchi et al. “GREEND: An energy consumption dataset of households in Italy and Austria”. In: *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE. 2014, pp. 511–516.

[44] Bishnu Nepal et al. “Electricity load forecasting using clustering and ARIMA model for energy management in buildings”. In: *Japan Architectural Review* 3.1 (2020), pp. 62–76.

[45] Anders Nilsson et al. “Smart homes, home energy management systems and real-time feedback: Lessons for influencing household energy consumption from a Swedish field study”. In: *Energy and Buildings* 179 (2018), pp. 15–25.

[46] Aaron Oord et al. “WaveNet: A Generative Model for Raw Audio”. In: (Sept. 2016). DOI: https://arxiv.org/abs/1609.03499.

[47] Aaron van den Oord et al. “WaveNet: A Generative Model for Raw Audio”. In: *CoRR* abs/1609.03499 (2016). arXiv: 1609.03499. URL: http://arxiv.org/abs/1609.03499.

[48] Yayu Peng et al. “Short-term load forecasting at different aggregation levels with predictability analysis”. In: *IEEE Innovative Smart Grid Technologies-Asia (ISGT Asia)*. 2019, pp. 3385–3390.

[49] Jairo Quirós-Tortós et al. “Statistical representation of EV charging: Real data analysis and applications”. In: *2018 Power Systems Computation Conference (PSCC)*. IEEE. 2018, pp. 1–7.

[50] Mina Razghandi et al. “Short-Term Load Forecasting for Smart Home Appliances with Sequence to Sequence Learning”. In: *IEEE International Conference on Communications (ICC)*. June 2021, pp. 1–6.

[51] Mina Razghandi et al. “Smart Home Energy Management: Sequence-to-Sequence Load Forecasting and Q-Learning”. In: *2021 IEEE Global Communications Conference (GLOBECOM)*. 2021, pp. 01–06. DOI: 10.1109/GLOBECOM46510.2021.9685380.

[52] Mina Razghandi et al. “Variational Autoencoder Generative Adversarial Network for Synthetic Data Generation in Smart Home”. In: *arXiv preprint arXiv:2201.07387* (Jan. 2022).

[53] Muhammad Sajjad et al. “A novel CNN-GRU-based hybrid approach for short-term residential load forecasting”. In: *Ieee Access* 8 (2020), pp. 143759–143768.

[54] Mike Schuster and Kuldip K Paliwal. “Bidirectional recurrent neural networks”. In: *IEEE transactions on Signal Processing* 45.11 (1997), pp. 2673–2681.

[55] Heng Shi, Minghao Xu, and Ran Li. “Deep learning for household load forecasting—A novel pooling deep RNN”. In: *IEEE Transactions on Smart Grid* 9.5 (2017), pp. 5271–5280.

[56] Elliott Skomski et al. “Sequence-to-sequence neural networks for short-term electrical load forecasting in commercial office buildings”. In: *Energy and Buildings* 226 (2020), p. 110350.

[57] Åse Lekang Sørensen et al. “Residential electric vehicle charging datasets from apartment buildings”. In: *Data in Brief* 36 (2021), p. 107105.

[58] Benjamin K Sovacool and Dylan D Furszyfer Del Rio. “Smart home technologies in Europe: A critical review of concepts, benefits, risks and policies”. In: *Renewable and sustainable energy reviews* 120 (2020), p. 109663.

[59] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. “Sequence to sequence learning with neural networks”. In: *Advances in Neural Information Processing Systems*. 2014, pp. 3104–3112.

[60] Lingling Tang, Yulin Yi, and Yuexing Peng. “An ensemble deep learning model for short-term load forecasting based on ARIMA and LSTM”. In: *IEEE SmartGridComm*. 2019, pp. 1–6.

[61] Chunming Tu et al. “Big data issues in smart grid–A review”. In: *Renewable and Sustainable Energy Reviews* 79 (2017), pp. 1099–1107.

[62] Sanna Tuomela et al. “Impacts of home energy management systems on electricity consumption”. In: *Applied Energy* 299 (2021), p. 117310.

[63] Akshay S.N. Uttama Nambi, Antonio Reyes Lua, and Venkatesha R. Prasad. “LocED: Location-Aware Energy Disaggregation Framework”. In: *ACM BuildSys’15*. 2015, pp. 45–54.

[64] SS Van Dam, CA Bakker, and JC Buiter. “Do home energy management systems make sense? Assessing their overall lifecycle impact”. In: *Energy Policy* 63 (2013), pp. 398–407.

[65] Xishun Wang, Minjie Zhang, and Fenghui Ren. “Learning customer behaviors for effective load forecasting”. In: *IEEE Transactions on Knowledge and Data Engineering* 31.5 (2018), pp. 938–951.

[66] Yi Wang et al. “Combining probabilistic load forecasts”. In: *IEEE Transactions on Smart Grid* 10.4 (2018), pp. 3664–3674.

[67] Yi Wang et al. “Review of smart meter data analytics: Applications, methodologies, and challenges”. In: *IEEE Transactions on Smart Grid* 10.3 (2018), pp. 3125–3148.

[68] Zhe Wang and Tianzhen Hong. “Generating realistic building electrical load profiles through the Generative Adversarial Network (GAN)”. In: *Energy and Buildings* 224 (2020), p. 110299.

[69] Zhe Wang and Tianzhen Hong. “Generating realistic building electrical load profiles through the Generative Adversarial Network (GAN)”. In: *Energy and Buildings* 224 (Oct. 2020), pp. 1–15.

[70] Shirantha Welikala et al. “Incorporating appliance usage patterns for non-intrusive load monitoring and load forecasting”. In: *IEEE Transactions on Smart Grid* 10.1 (2017), pp. 448–461.

[71] Charlie Wilson, Tom Hargreaves, and Richard Hauxwell-Baldwin. “Benefits and risks of smart home technologies”. In: *Energy Policy* 103 (2017), pp. 72–83.

[72] Di Wu et al. “Multiple Kernel Learning based Transfer Regression for Electric Load Forecasting”. In: *IEEE Transactions on Smart Grid* 11.2 (2020), pp. 1183–1192.

[73] Xu Xu et al. “A Multi-Agent Reinforcement Learning-Based Data-Driven Method for Home Energy Management”. In: *IEEE Trans. on Smart Grid* 11 (July 2020), pp. 3201–3211.

[74] Ke Yan et al. “A Hybrid LSTM Neural Network for Energy Consumption Forecasting of Individual Households”. In: *Ieee Access* 7 (2019), pp. 157633–157642.

[75] Yandong Yang et al. “Bayesian Deep Learning-Based Probabilistic Load Forecasting in Smart Grids”. In: *IEEE Transactions on Industrial Informatics* 16.7 (2019), pp. 4703–4713.

[76] Yang Yang and Shu Wang. “Resilient residential energy management with vehicle-to-home and photovoltaic uncertainty”. In: *International Journal of Electrical Power & Energy Systems* 132 (2021), p. 107206.

[77] Leehter Yao, Zolboo Damiran, and Wei Hong Lim. “Energy management optimization scheme for smart home considering different types of appliances”. In: *2017 IEEE international conference on environment and electrical engineering and 2017 IEEE industrial and commercial power systems Europe (EEEIC/I&CPS Europe)*. IEEE. 2017, pp. 1–6.

[78] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. “Time-series generative adversarial networks”. In: *Advances in Neural Information Processing Systems* 32 (2019).

[79] Mojtaba Yousefi et al. “Predictive home energy management system with photovoltaic array, heat pump, and plug-in electric vehicle”. In: *IEEE Transactions on Industrial Informatics* 17.1 (2020), pp. 430–440.

[80] Chun-Nam Yu, Piotr Mirowski, and Tin Kam Ho. “A Sparse Coding Approach to Household Electricity Demand Forecasting in Smart Grids”. In: *IEEE Transactions on Smart Grid* 8.2 (2016), pp. 738–748.

[81] Chaoyun Zhang et al. “Sequence-to-point learning with neural networks for non-intrusive load monitoring”. In: *arXiv* (2016). URL: https://arxiv.org/abs/1612.09106.

[82] Chi Zhang et al. “Generative Adversarial Network for Synthetic Time Series Data Generation in Smart Grids”. In: *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. 2018, pp. 1–6. DOI: 10.1109/SmartGridComm.2018.8587464.

[83] Chi Zhang et al. “Generative adversarial network for synthetic time series data generation in smart grids”. In: *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE. 2018, pp. 1–6.

[84] Jian Zheng et al. “Electric Load Forecasting in Smart Grids Using Long-Short-Term-Memory based Recurrent Neural Network”. In: *in Proc. of 2017 51st Annual Conference on Information Sciences and Systems (CISS)*. Mar. 2017, pp. 1–6.

[85] Bin Zhou et al. “Smart home energy management systems: Concept, configurations, and scheduling strategies”. In: *Renewable and Sustainable Energy Reviews* 61 (2016), pp. 30–40.

[86] Hao Zhou and Melike Erol-Kantarci. “Correlated deep q-learning based microgrid energy management”. In: *2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. IEEE. 2020, pp. 1–6.

[87] Hao Zhou and Melike Erol-Kantarci. “Decentralized microgrid energy management: A multi-agent correlated q-learning approach”. In: *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE. 2020, pp. 1–7.

[88] Hao Zhou et al. “Multi-agent Bayesian Deep Reinforcement Learning for Microgrid Energy Management under Communication Failures”. In: *IEEE Internet of Things Journal* (2021).