UNLOCKING THE POTENTIAL OF AI IN ENERGY UTILIZATION PROGNOSIS

Jui-Lien Hsia & Anujraaj Gopalsamy Sakthivel

Abstract. — The forecasting of residential electricity demand has become a critical aspect in the development of sustainable energy systems. Accurate prediction of electricity consumption patterns is essential in order to maintain a stable energy supply and prevent overloading the grid. The present study aimed to explore the effectiveness of the Stride TCN model in forecasting electricity demand using data obtained from a Raspberry Pi. The collected data underwent a rigorous preprocessing stage, which involved transforming the raw time series data into a format suitable for training the model. Through the use of the Stride TCN model, the study was able to demonstrate the potential of deep learning models in predicting residential electricity demand with high accuracy. This research sheds light on the importance of incorporating cutting-edge machine learning techniques in energy forecasting and highlights the significance of precise demand forecasting in the development of a sustainable energy future.

Contents

Key words and phrases. — Forecast, Electricity Demand, Data Collection, Preprocessing, and Stride TCN.

1. Introduction

The need for accurate forecasting of energy demand in residential buildings has become increasingly important in recent years, particularly with the growing emphasis on energy efficiency and the integration of smart building technologies [IEA23]. In particular, the accurate prediction of electric energy demand in residential buildings has become a crucial aspect of managing energy consumption and reducing costs. The French residential sector has been identified as a key area of focus, as the country seeks to meet its ambitious energy efficiency and carbon reduction goals. In light of this, the application of machine learning models for forecasting energy demand in residential buildings in France has been the subject of a growing body of research. Despite the promising results that have been achieved thus far, there remains a need for further analysis and development in this field [SNG19].

This article aims to provide an in-depth examination of the current state of the art in the use of machine learning models for forecasting energy demand in residential buildings, with a particular focus on the prediction of electric energy demand. By highlighting the challenges and opportunities presented by this area of research, it is hoped that this article will inform future developments in the field and contribute to the advancement of smart building technologies.

2. State of the Art

To date, the majority of research on deep learning (DL) models for forecasting energy loads in residential buildings has focused solely on electric loads. This limited scope may be attributed to the additional time and resources required for data acquisition of non-electrical loads such as thermal, lighting, and natural gas. Additionally, forecasting energy loads at a granular level can be challenging due to the inherent uncertainty and volatility of these loads [BBO18].

A variety of DL-based models have been applied at the residential level, including convolutional neural networks (CNNs), long short-term memory (LSTM) networks, and comparison-based models as highlighted in recent publications. Of these, LSTM-based models have achieved the most widespread implementation and have demonstrated promising results. For example, Amarasignhe et al. [AMM17] compared DL techniques such as CNNs, LSTMs, and factorized restricted Boltzmann machines (FRBMs) to traditional machine learning (ML) techniques such as artificial neural networks (ANNs) and support vector machines (SVMs) for forecasting electric demand in a building. The forecast horizon was 60 hours ahead and the study utilized four years of training data. The results of this study showed that DL-based forecasting models obtained lower forecasting errors compared to standard ML-based techniques, with LSTMs achieving the smallest error.

Similarly, Hossen et al. [Hos+19] conducted a preliminary analysis comparing simple recurrent neural networks (RNNs), gated recurrent units (GRUs), and LSTMs to other data-driven approaches such as autoregressive integrated moving average (ARIMA), generalized linear models (GLMs), random forests (RFs), SVMs, and feedforward neural networks (FFNNs). The results of this study indicated that all RNN-based models achieved superior forecasting performance compared to other data-driven models.

CNNs have also shown promising results in their applications to residential energy forecasting models. For instance, Estebsari et al. [EE220] compared CNN-based models to ANNs and SVMs for a residential house. The results of this study showed that CNN-based models achieved higher accuracy than SVM and ANN models. However, despite these promising results, the application of DL-based techniques for residential energy load forecasting remains an area with a relatively small sample size of applications and further research is required to fully understand its potential [RZ21].

In recent times, a specialized architecture known as temporal convolutional networks (TCN) has gained popularity as a suitable method for handling time series data. The concept of TCN was first introduced in [BKK18], where it was compared to several recurrent neural networks (RNNs) for sequence modeling tasks. TCNs utilize causal dilated convolution to enable the capturing of longer-term dependencies while preventing information loss. Additionally, they possess several benefits over RNNs such as reduced memory requirements, parallel processing of long sequences, and a more stable training procedure. Several studies have successfully employed TCNs for time series forecasting tasks, including an architecture using stacked dilated convolutions proposed in [BBO18] to enhance the performance of long short-term memory networks for financial domain problems, an encoder-decoder deep TCN proposed in [Che+20] for multiple related time series in the sales domain, and a multivariate time series forecasting model for meteorological data proposed in [Wan+19] which outperformed several popular deep learning models. However, to the best of our knowledge, the potential of TCNs for univariate time series forecasting problems related to electricity demand data has yet to be explored [Lar+20].

Temporal Convolutional Networks (TCNs) are a class of deep learning models specifically designed for time series forecasting. They are built on the architecture of Convolutional Neural Networks (CNNs) and are well-suited for applications involving sequential data such as energy demand forecasting.

The key advantage of TCNs is their ability to effectively capture long-term dependencies in time series data. Traditional Recurrent Neural Networks (RNNs) such as Long Short-Term Memory (LSTM) networks have difficulty in capturing dependencies beyond a certain horizon due to the vanishing gradient problem. TCNs, on the other hand, use dilated convolutions to increase the receptive field of the network without increasing the number of parameters. This allows TCNs to capture temporal patterns at multiple scales, effectively addressing the problem of long-term dependencies in time series data.

In TCNs, the input data is passed through a series of convolutional layers, each followed by a non-linear activation function. The convolutional layers use dilated filters, which means that the filters are spaced apart by a certain factor. This allows the network to increase its receptive field without increasing the number of parameters. The output of the last convolutional layer is passed through a fully connected layer, which produces the final forecast.

Thus making TCNs a powerful class of deep learning models that are well-suited for time series forecasting applications such as energy demand forecasting. Their ability to effectively capture long-term dependencies in sequential data makes them a strong candidate for addressing the challenges of forecasting in the energy sector. With the increasing availability of high-resolution energy data, TCNs have the potential to significantly improve the accuracy and reliability of energy forecasting.

3. Methodology

Analysis only make sense if we are having a right data and we understand it. Therefore, before doing analysis, we clean and explore the raw data. We dropped the repeated data and sampled the data by 1 hour. After obtaining a reasonable stationary profile for the label and features, the data are combined into one dataset and only the hours with complete data are extracted.

With the clean data, we explored the data and investigate the importance of each feature to the label, which in our case is the electricity consumption.

3.1. Data Collection

The data is collected from the Expe-smarthouse project which is an initiative aimed at providing researchers with access to real-time data from a 120 square meter household [Jer]. This smart home is equipped with approximately 340 measuring points, which are made available through a Grafana portal that utilizes an Influxdb database. The data collected includes measures of electricity, gas, and water consumption, temperature, humidity, and brightness levels, as well as information on door and window openings, motion sensor readings, lighting states, and air quality in each room. Additionally, data on outdoor weather conditions is also collected.

The process of collecting sensor data using Raspberry Pi and storing it in a database is commonly referred to as data logging. One popular method for data logging is to use a Raspberry Pi in conjunction with a sensor such as a power meter to measure electricity consumption and weather sensors to measure temperature and humidity. The data is then collected and stored in a database, such as InfluxDB, which allows for easy retrieval and analysis.

To collect data, a Raspberry Pi can be configured with the necessary software and libraries to communicate with the sensors. The sensor data is then read by the Raspberry Pi at regular intervals and sent to the InfluxDB database via a protocol such as the InfluxDB Line Protocol. The data is then stored in a structured format, allowing for easy querying and analysis. In addition, the data can be visualized using a tool such as Grafana for further analysis.

3.2. Data Analysis

The data obtained as CSV files are read as data frames and their quality is checked in terms of resolution and time period covered. The data are then resampled into hourly resolution and concatenated together. In order to understand the dependency between the electricity meter data and various weather data the following correlation matrix is plotted as shown in the *Figure 1 in Page 6*. As it is evident there is no strong correlation between the electricity consumption and weather as the buildings heating system is not electric. Thus the weather data is dropped from focus for the moment.

Exploratory data analysis (EDA) is a crucial step in the process of time series forecasting. This step involves examining the characteristics and patterns of the data in order to gain insights and identify any potential issues or outliers.

One common technique used in EDA of time series data is visual inspection. The electric power time series is plotted as shown in the figure. This can be done by plotting the data over time and observing any trends, seasonality, or other patterns that may be present as shown in the *Figure 2 in Page 6*. Additionally, summary statistics such as mean, median, and standard deviation can be calculated to provide a quick overview of the distribution of the data as shown in the *Table 1 in Page 7*.

Another technique used in EDA of time series data is decomposition. This involves breaking down the time series data into its component parts, such as trend, seasonal, and residual components. Decomposition can provide further insights into the underlying patterns in the data and can help identify any potential issues that may need to be addressed before using machine learning techniques for forecasting.

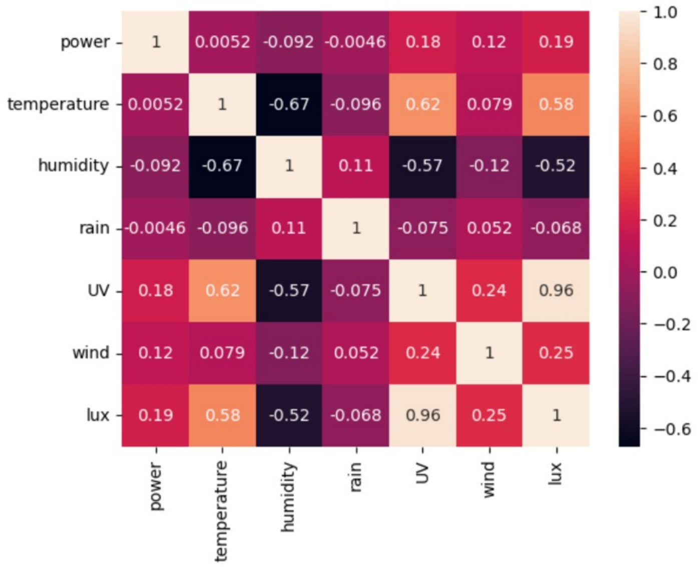

FIGURE 1. Correlation Heatmap

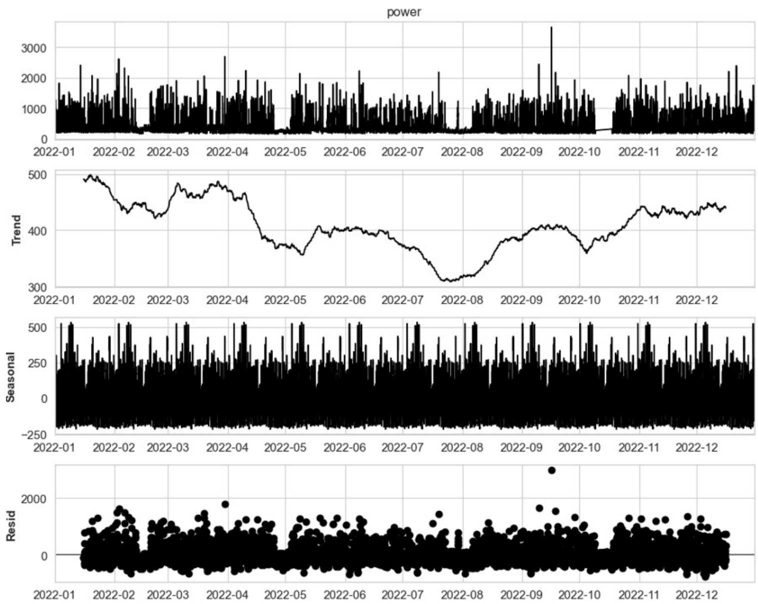

FIGURE 2. Time Series Decomposition

TABLE 1. Statiscal Analysis of Electricity Consumption Data.

count	8736.000000
mean	412.358355
std	305.836092
min	151.689498
25%	222.337282
50%	294.822410
75%	446.785882
max	3659.466292

TABLE 2. Augmented Dickey Fuller Test

ADF Statistic	-12.38
n_lags	5.02e-23
p-value	5.02e-23
Critical Values 1%	-3.43
Critical Values 5%	-2.86
Critical Values 10%	-2.56

Finally, statistical tests such as the Augmented Dickey-Fuller test can be used to determine if the time series data is stationary, which is an assumption of many machine learning models as shown in the *Table 2 in Page 7*. If the data is not stationary, then a transformation such as differencing, or a seasonal decomposition may be necessary.

Overall, the goal of exploratory data analysis is to gain a deeper understanding of the data before using machine learning techniques to forecast energy demand, and to identify any potential issues that may need to be addressed in order to improve the accuracy of the forecasts.

3.3. Stride - TCN Model

Temporal Convolutional Networks (TCNs) have gained popularity as a specialized architecture for dealing with time series data. The architecture of a TCN, as shown in the *Figure 3 in Page 8* consists of multiple components, including input tensor, residual blocks, basic layers, weight normalization, ReLU activation, dropout, and causal convolutional layers, all of which play an important role in processing time series data [Fra20] [Anh+22].

The input tensor is fed into the TCN, which represents the time series data. The input tensor is then processed by a series of residual blocks, which are the building blocks of the TCN. The basic layer of the residual block takes in the input tensor, and applies weight normalization followed by a ReLU activation

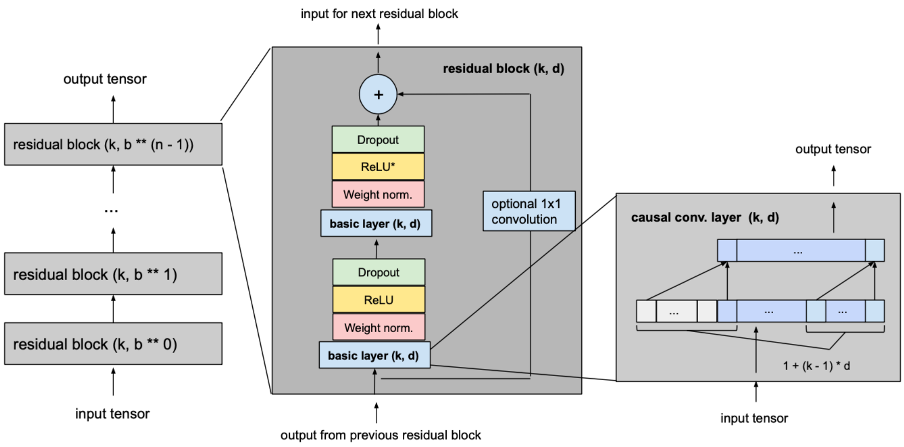

FIGURE 3. TCN Architecture [Fra20]

function. The output of the basic layer is then added to the original input tensor, and this result is referred to as the residual.

The residual is then passed through a dropout layer, which helps in preventing overfitting by randomly zeroing out some of the activations. The output of the dropout layer is then processed by the next residual block.

A casual convolutional layer, which is a convolutional layer with a filter that only processes past information, is applied to the output of the residual block. This layer extracts relevant features from the time series data, allowing the TCN to capture long-term dependencies. The final output of the TCN is a tensor that represents the processed time series data. The output tensor is then used for the task of time series forecasting.

Stride-TCN is a variant of the Temporal Convolutional Network (TCN) architecture for processing sequential data. The main difference between the traditional TCN and Stride-TCN is the usage of strided convolutions in the latter, which are convolutional layers with a stride greater than 1, effectively reducing the spatial resolution of the input data. The purpose of this reduction is to reduce the number of parameters in the model, speed up computation, and prevent overfitting. Stride-TCN is particularly useful for handling large sequence lengths, such as long-term time series data, where a traditional TCN might be computationally infeasible. The performance of Stride-TCN has been shown to be competitive with traditional TCN models in a range of sequential data processing tasks, including time series forecasting and activity recognition.

A Stride-TCN is employed in this project to forecast the electricity demand. The model is obtained from the Darts library and the main parameters of the model include,

$$\text{input chunk length} = 7*24 \text{ hours}$$
$$\text{output chunk length} = 1 \text{ hour}$$

The Mean Absolute Error (MAE) is a commonly used evaluation metric in time series forecasting to measure the difference between the predicted and actual values. When evaluating the performance of the Stride-TCN model for electricity demand forecasting, the MAE can be calculated by taking the average absolute difference between the predicted and actual demand values over a specific time period. Mathematically, this can be represented as follows:

$$\text{MAE} = |(\mathrm{y}_i - y_p)|/n$$

Where,
n = number of observations
$\mathrm{y}_i = actualelectricitydemandvalues$
$\mathrm{y}_p = predictedelectricitydemandvalues$

The lower the value of MAE, the better the performance of the model. A lower MAE value indicates that the model has a higher accuracy in predicting the electricity demand values. In other words, the predictions are closer to the actual values, making the model more reliable for forecasting purposes.

4. Results & Discussions

This section discusses the model architecture used and results obtained.

4.1. Hourly Resolution

The parameters in a stride TCN model [Uni20] can be explained as follows:

1. input_chunk_length: This parameter specifies the length of the input chunk for the TCN model. A value of 7 x 24 means that the input chunk is of length 7 days and 24 hours, i.e., 168 hours.
2. output_chunk_length: This parameter specifies the length of the output chunk. A value of 1 means that the output chunk is of length 1 hour.
3. n_epochs: This parameter specifies the number of training epochs for the model. A value of 50 means that the model will be trained for 50 iterations.
4. dropout: This parameter specifies the dropout rate for the model, which is used to prevent overfitting. A value of 0.2 means that 20% of the neurons will be dropped during each iteration of training.

5. dilation_base: This parameter specifies the base dilation rate for the causal convolutional layer. A value of 2 means that the dilation rate will be a power of 2.
6. weight_norm: This parameter specifies whether weight normalization should be used for the model. A value of True means that weight normalization will be used.
7. kernel_size: This parameter specifies the size of the kernel used in the causal convolutional layer. A value of 5 means that a 5 x 5 kernel will be used.
8. num_filters: This parameter specifies the number of filters used in the causal convolutional layer. A value of 8 means that 8 filters will be used.
9. nr_epochs_val_period: This parameter specifies the number of epochs after which the model will be validated. A value of 1 means that the model will be validated after every 1 epoch.
10. random_state: This parameter specifies the random seed used for reproducibility. A value of 0 means that the model will be initialized with the same random seed each time it is run.

The historical_forecasts function is used to generate historical forecasts based on the trained TCN model. The parameters for this function are:

1. series: This parameter specifies the time series data that is to be used for forecasting.
2. start: This parameter specifies the start point of the time series data to be used for forecasting. A value of 0.7 means that 70% of the data will be used for training and the remaining 30% will be used for validation.
3. forecast_horizon: This parameter specifies the number of steps ahead the model will forecast. A value of 1 means that the model will forecast 1 step ahead.
4. stride: This parameter specifies the step size for the forecast. A value of 1 means that the forecast will be made for every step in the time series.
5. retrain: This parameter specifies whether the model should be retrained for each forecast. A value of False means that the model will not be retrained.

In a scenario where the hourly electricity consumption data is being analyzed, a Stride TCN model is trained with 80% of the available data, with the remaining 20% of the data being reserved for testing purposes. The model is trained with the objective of accurately forecasting the electricity demand. To evaluate the performance of the model, mean absolute error (MAE) is used as the evaluation metric. The final results as seen in *Figure 4 in Page 11* show that the Stride TCN model performs with an MAE of 0.0416, which indicates that, on average, the model's forecasts deviate from the actual values by 0.0416 units. This demonstrates the efficacy of the Stride TCN model in forecasting electricity demand, with a low deviation between the predicted values and the actual values.

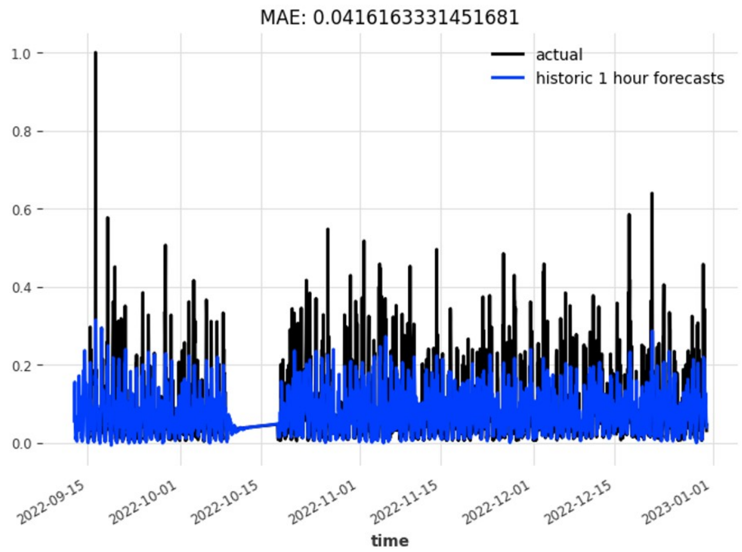

FIGURE 4. Model Prediction Evaluation - Hourly Resolution Input Data

4.2. Daily Resolution

When the same approach of using the stride TCN model is repeated on electricity consumption data collected at a daily resolution, the results obtained are not as favorable as in the previous scenario. The model is trained using 80% of the data, with the remaining 20% of the data being used for testing. The evaluation of the model is done using the Mean Absolute Error (MAE) metric. The obtained MAE score is 0.1465 as shown in the *Figure 5 in Page 12*. This value indicates that there is a significant difference in the performance of the model when it is trained on data collected at different temporal resolutions. It is important to note that the higher the resolution of the data, the more complex the model's task becomes, as it needs to take into account a greater amount of information in order to make accurate predictions. In this scenario, the model's performance might not have been optimal due to the presence of additional information in the daily resolution data, which the model may have struggled to fully process and incorporate into its predictions.

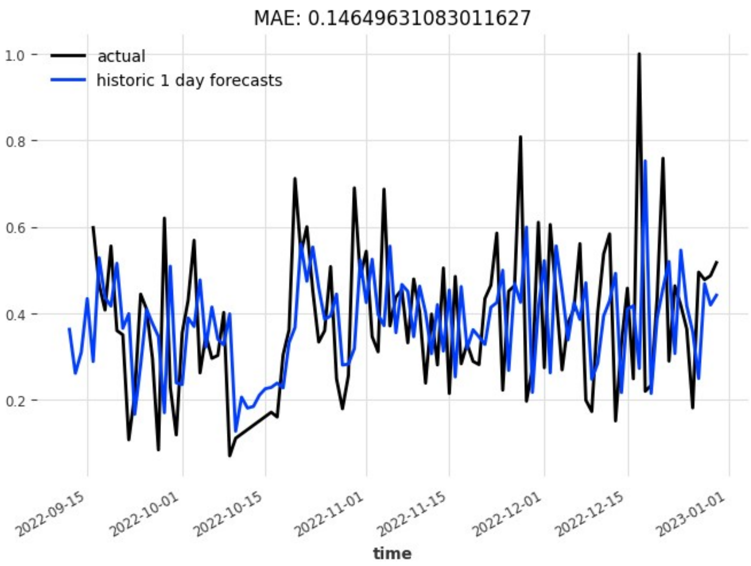

FIGURE 5. Model Prediction Evaluation - Daily Resolution Input Data

5. Conclusions & Reflections

In conclusion, this machine learning lab project, utilizing the Stride TCN model on electricity demand data collected from a Raspberry Pi device, provided a valuable opportunity for learning and application of various concepts and techniques. Through the course of this project, it has become evident that data collection and preprocessing play crucial roles in determining the success of a machine learning model. The process of transforming raw data into a format suitable for modeling, identifying and handling missing values, and normalizing the data, among other steps, must be carefully executed in order to obtain accurate results.

The Stride TCN model was trained on 80% of the hourly electricity consumption data and tested on the remaining 20% of the data, with a mean absolute error score of 0.0416 obtained. The same procedure was repeated on a daily resolution data, resulting in a mean absolute error score of 0.1465.

While this project has provided a solid foundation for understanding the Stride TCN model and its applications, there is room for improvement and further exploration. For example, the parameters of the model can be further optimized through hyperparameter tuning, to improve its performance and accuracy. Additionally, incorporating other relevant variables, such as weather data or socioeconomic indicators, into the analysis may also lead to more accurate results.

Overall, this lab project has been a valuable experience and the learnings gained will be useful in future endeavors. It has demonstrated the importance of careful data collection and preprocessing, and the potential of the Stride TCN model in forecasting electricity demand. The future prospects of this project are promising, and further exploration and experimentation with the Stride TCN model and other time series forecasting techniques are encouraged.

6. Code

The code and the associated data can be found in this GitHub repository.

References

[AMM17] Kasun Amarasinghe, Daniel L. Marino, and Milos Manic. "Deep neural networks for energy load forecasting". In: *IEEE International Symposium on Industrial Electronics* (Aug. 2017), pp. 1483–1488. DOI: 10.1109/ISIE.2017.8001465.

[Anh+22] Le Hoang Anh et al. "Stride-TCN for Energy Consumption Forecasting and Its Optimization". In: *Applied Sciences 2022, Vol. 12, Page 9422* 12.19 (Sept. 2022), p. 9422. ISSN: 2076-3417. DOI: 10.3390/APP12199422. URL: https://www.mdpi.com/2076-3417/12/19/9422/htm%20https://www.mdpi.com/2076-3417/12/19/9422.

[BKK18] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling". In: (Mar. 2018). URL: http://arxiv.org/abs/1803.01271.

[BBO18] Anastasia Borovykh, Sander Bohte, and Cornelis W. Oosterlee. "Dilated convolutional neural networks for time series forecasting". In: *Journal of Computational Finance* (2018). ISSN: 14601559. DOI: 10.21314/JCF.2019.358.

[Che+20] Yitian Chen et al. "Probabilistic forecasting with temporal convolutional neural network". In: *Neurocomputing* 399 (July 2020), pp. 491–501. ISSN: 18728286. DOI: 10.1016/J.NEUCOM.2020.03.011.

[EE220] A Estebsari, R Rajabi - Electronics, and undefined 2020. "Single residential load forecasting using deep learning and image encoding techniques". In: *mdpi.com* 9.1 (Jan. 2020). DOI: 10.3390/electronics9010068. URL: https://www.mdpi.com/607976.

[Fra20] Francesco Lässig. *Temporal Convolutional Networks and Forecasting*. 2020. URL: https://unit8.com/resources/temporal-convolutional-networks-and-forecasting/.

[Hos+19] Tareq Hossen et al. "Residential Load Forecasting Using Deep Neural Networks (DNN)". In: *2018 North American Power Symposium, NAPS 2018* (Jan. 2019). DOI: 10.1109/NAPS.2018.8600549.

[IEA23] IEA. *Buildings*. 2023. URL: https://www.iea.org/topics/buildings.

[Jer] Jerome Ferrari. *EXPE-SMARTHOUSE – The full connected living house*. URL: http://expe-smarthouse.duckdns.org/?page_id=7&lang=en.

[Lar+20] Pedro Lara-Benítez et al. "Temporal Convolutional Networks Applied to Energy-Related Time Series Forecasting". In: *Applied Sciences 2020, Vol. 10, Page 2322* 10.7 (Mar. 2020), p. 2322. ISSN:

2076-3417. DOI: 10.3390/APP10072322. URL: https://www.mdpi.com/2076-3417/10/7/2322/htm%20https://www.mdpi.com/2076-3417/10/7/2322.

[RZ21] Jason Runge and Radu Zmeureanu. "A Review of Deep Learning Techniques for Forecasting Energy Use in Buildings". In: *Energies 2021, Vol. 14, Page 608* 14.3 (Jan. 2021), p. 608. ISSN: 1996-1073. DOI: 10.3390/EN14030608. URL: https://www.mdpi.com/1996-1073/14/3/608/htm%20https://www.mdpi.com/1996-1073/14/3/608.

[SNG19] Ljubisa Sehovac, Cornelius Nesen, and Katarina Grolinger. "Forecasting building energy consumption with deep learning: A sequence to sequence approach". In: *Proceedings - 2019 IEEE International Congress on Internet of Things, ICIOT 2019 - Part of the 2019 IEEE World Congress on Services* (July 2019), pp. 108–116. DOI: 10.1109/ICIOT.2019.00029.

[Uni20] Unit8. *Temporal Convolutional Network — darts documentation*. 2020. URL: https://unit8co.github.io/darts/examples/05-TCN-examples.html#Daily-energy-production.

[Wan+19] Renzhuo Wan et al. "Multivariate temporal convolutional network: A deep neural networks approach for multivariate time series forecasting". In: *Electronics (Switzerland)* 8.8 (Aug. 2019). ISSN: 20799292. DOI: 10.3390/ELECTRONICS8080876.

MLO Lab Report, Dated 29 January 2023

JUI-LIEN HSIA & ANUJRAAJ GOPALSAMY SAKTHIVEL,
Grenoble INP-Ense3,
21 Av. des Martyrs, 38031 Grenoble
E-mail : Jui.Lien@student.innoenergy.com,
Anujraaj.Gopalsamy@student.innoenergy.com