TUNISIAN REPUBLIC
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

UNIVERSITY OF CARTHAGE

HIGHER SCHOOL OF COMMUNICATIONS OF TUNIS

REPORT

TUTORED PROJECT 2024

Data Augmentation Strategy Using GAN for Load Disaggregation

Realized by

BEN AMOR ANAS & KTARI AYMEN

HOST COMPANY :

Sagemcom

Supervized By
CHAABANE FERDAOUAS & MAALEJ ASMA

Year :
2023-2024

Acknowledgments

The completion of this project is the result of a collective effort from many individuals who generously shared their guidance and expertise.

We would like to express my deepest gratitude to **Mrs. Ferdaouas CHAABANE** for her continuous support, availability, and mentorship throughout this internship, as well as for making this project feels like a true team endeavor.

We also extend our sincere thanks to **Mrs. Asma MAALEJ** for her valuable contribution and support during this project.

We extend our sincere thanks to everyone at the **Sagemcom** team for making this internship a rewarding experience and for providing a work environment that fostered confidence and growth.

Lastly, we address our heartfelt thanks to the **jury members** who honored us by evaluating our work and enhancing it with their valuable feedback.

Finally, we warmly thank **Sup'Com** for its support in both our professional and personal development.

Contents

General Introduction **1**

1 Scope Statement **2**
1.1 Hosting Company . . . 2
1.1.1 General Overview . . . 2
1.1.2 Offered Services . . . 2
1.2 Presentation of the Internship Objective . . . 3
1.2.1 Project context . . . 3
1.2.2 Problem . . . 3
1.2.3 Proposed Solution . . . 3

2 Theoretical Background **4**
2.1 Time series . . . 4
2.1.1 Definition . . . 4
2.1.2 Characteristics . . . 4
2.2 Sliding Window . . . 4
2.3 Generative Adversarial Networks . . . 4
2.3.1 GANs Input . . . 5
2.3.2 Generator . . . 5
2.3.3 Discriminator . . . 6
2.3.4 Self-Attention Mechanism . . . 7

3 Implementation and Results **9**
3.1 Data Analysis and Preprocessing . . . 9
3.1.1 Dataset Overview . . . 9
3.1.2 Data Synchronization and Aggregation . . . 10
3.1.3 Data Preprocessing . . . 10
3.1.3.1 Sliding Window Technique . . . 10
3.1.3.2 Data Cleaning and Normalization . . . 10
3.1.3.3 Segmenting the Data . . . 11
3.1.3.4 Training Data Preparation . . . 11
3.2 GAN Training Process . . . 11
3.2.1 Training Parameters . . . 11
3.3 Synthetic Data Integration . . . 12
3.3.1 Synthetic Signal Visualizations . . . 12
3.4 Results . . . 12
3.4.1 Performance Metrics . . . 12
3.4.2 Comparison Before and After Augmentation . . . 13
3.4.3 Challenges and Limitations . . . 14

Conclusion **15**

Bibliographie **16**

List of Figures

1.1 Logo Sagemcom 2

2.1 GAN Architecture 5
2.2 Generator Architecture 6
2.3 Discriminator Architecture 7
2.4 Multi-head Self-Attention Mechanism 8

3.1 Real Microwave Active Power Consumption Signal samples. 12
3.2 Synthetic Microwave Active Power Consumption Signal samples. 12
3.3 Comparison of Real vs. Synthetic Microwave Active Power Consumption Signals. 12
3.4 Loss and MAE before Augmentation 13
3.5 Loss and MAE after Augmentation 13
3.6 Comparison of Predictions vs Ground Truth before and after augmentation. . . . 14

List of Tables

3.1 REDD Dataset Specifications 9
3.2 GAN Training Hyperparameters 11
3.3 Performance Metrics Before and After Augmentation 13

List of Abbreviations

GANs : Generative Adversarial Networks

NILM : Non-Intrusive Load Monitoring

Conv1D : 1-Dimensional Convolutional Layer

LeakyReLu : Leaky Rectified Linear Unit

CSV : Comma-Separated Values

REDD : Residential Energy Disaggregation Data Set

MAE : Mean Absolute Error

MSE : Mean Squared Error

RETE : Relative Error of Total Energy

General Introduction

In the evolving landscape of energy management, accurate monitoring of electricity consumption is critical for optimizing energy efficiency and reducing waste. One of the most promising techniques in this domain is load disaggregation NILM, which refers to the process of separating the total energy consumption of a building or system into individual appliance-level usage. This provides valuable insights into the energy behavior of various devices, enabling both consumers and energy providers to make more informed decisions on energy use and conservation.

However, the implementation of load disaggregation faces significant challenges due to the complexities of electrical signals, limited training data, and noisy environments. To overcome these limitations, this project explores the application of data augmentation as a strategy to enhance the performance of machine learning models used for load disaggregation. Data augmentation techniques can artificially expand the training datasets, providing the models with a broader range of scenarios and improving their ability to generalize across diverse operating conditions.

The main objective of this project is to develop and test a data augmentation solution that can effectively boost the accuracy and robustness of load disaggregation models, especially when implemented on embedded microprocessors. The solution aims to evaluate the impact of data augmentation on the model's performance and complexity, ensuring that the final implementation can be deployed in real-world environments where resource constraints are often a concern.

The report begins by setting the context for the project, outlining the hosting company, the objectives, and the problem being addressed. The second chapter provides an overview of the theoretical background that covers the theoretical foundations of our project, focusing on GANs, which are central to our data augmentation approach.

Chapter three focuses on the implementation process, specifically the preprocessing of data and model evaluation. It will examine how factors like measurement frequency and data volume influence model performance.

The final section of the chapter will present the results of the load disaggregation model, with a particular emphasis on the effects of data augmentation using GANs. Additionally, it will discuss potential improvements and future research directions.

In conclusion, the report will summarize the key findings and propose avenues for further study.

Chapter 1

Scope Statement

Introduction

This chapter provides an overview of the project. We begin by introducing the hosting company, followed by a presentation of the project's general context.

1.1 Hosting Company

1.1.1 General Overview

SAGEMCOM [1] is a French group with a presence in over 40 countries worldwide. With 40 years of experience in the telecommunications industry, it is the European leader in the market for high-value-added communicating terminals such as decoders, internet boxes, electric meters, and connected objects. The group has a turnover of 3 billion euros and employs 6,500 people across more than 50 countries. SAGEMCOM designs, manufactures, and delivers more than 31.5 million terminals globally. The group operates in several key sectors, including:

- Broadband (decoders, TV/Internet boxes)
- Smart City (smart meters)
- Internet of Things (connected objects)

SAGEMCOM aims to become one of the world's leaders in high-value-added communicating terminals by focusing on dynamic markets. The company strives to be the first to offer its customers products that integrate the latest technological innovations.

Figure 1.1 – Logo Sagemcom

1.1.2 Offered Services

SAGEMCOM Software and Technologies (SST), a subsidiary of SAGEMCOM, is a global R&D center founded in 2005, with locations in key countries like France, the U.S., and China. SST designs and develops software solutions for the digital transmission and high-speed communication market, with three main activity areas: Digital Television Decoders (ADT), Residential Terminals (ATR), and Energy & Telecom (AET). The company focuses on providing

tailored solutions for public services and telecom operators. To enhance competitiveness and enter new markets, SAGEMCOM is exploring AI for automotive interior monitoring systems, aiming to develop intelligent product solutions through proof of concepts (POC).

1.2 Presentation of the Internship Objective

1.2.1 Project context

As the global demand for energy continues to rise, concerns regarding sustainability and the increasing cost of energy become more pronounced. Optimizing energy consumption is critical, especially in light of limited non-renewable resources. Load disaggregation, or NILM [2], plays a pivotal role in this optimization process by disaggregating total energy consumption into the usage of individual appliances. This not only provides valuable insights into energy efficiency but also facilitates better management and allocation of energy resources across households and industries.

1.2.2 Problem

A significant challenge in energy management tasks, such as load forecasting and anomaly detection, lies in the complexity of time-series data. The intricate patterns found in energy consumption often reflect long-term cycles and seasonality, which can vary depending on multiple factors, such as occupancy behavior and appliance efficiency. Compounding this complexity is the issue of data imbalance. Frequently used appliances, such as refrigerators, generate abundant data, while infrequently used appliances, like ovens and microwaves, provide limited data. This imbalance makes it difficult for models to learn accurate energy consumption patterns,leading to incorrect predictions and lower accuracy in load disaggregation models. The absence of sufficient data for these infrequent appliances further undermines model performance, hindering the effective analysis and management of energy resources.

1.2.3 Proposed Solution

To tackle the issue of insufficient data for load disaggregation, we suggest using data augmentation strategy [3] that we implemented using Generative AI, specifically GANs [4] , for data augmentation. GANs are well-suited for this task as they can generate realistic synthetic data that closely resembles actual appliance energy usage patterns. By producing diverse and balanced data, GANs help fill in the gaps where real data is scarce, particularly for less frequently used appliances. This augmented dataset improves model performance, making the models more accurate and adaptable to a wider range of scenarios, ultimately leading to more effective load disaggregation and better energy management.

Conclusion

This chapter provides an overview of the hosting company, as well as the project details and the internship objectives. The next chapter will focus on the theoretical background relevant to the project.

Chapter 2

Theoretical Background

Introduction

In this chapter, we will explore the theoretical aspects of our project, with a particular focus on the GANs architecture. Since our project aims to augment data, understanding GANs is essential as they form the foundation of our approach.

2.1 Time series

2.1.1 Definition

Time series [5] consist of observations collected over time, capturing how dependencies evolve. Unlike other data types, time series data reveals how values change and influence one another. It often exhibits patterns like trends or seasonal variations that need to be considered in analysis. A large dataset ensures consistency, filters noise, and helps identify genuine trends. Time series analysis is valuable for prediction, as it enables forecasting future values based on historical data.

2.1.2 Characteristics

A time series includes **trend** (long-term increase or decrease), **seasonality** (repeated patterns), **residual** (remaining data after trend and seasonality), **cycle** (irregular trends), and **stationarity** (constant statistical properties over time). These elements are essential for analyzing and forecasting data.

2.2 Sliding Window

The sliding window [6] technique splits time series into short, consistent segments for energy load disaggregation. It transforms raw data into a format suitable for machine learning.

Sequences are cut into fixed-length segments, then converted to 2D tensors, while the output is 1D. This method boosts efficiency by reducing operations needed for sequential data.

2.3 Generative Adversarial Networks

GANs are a Generative AI models used for generating new data that mimics the distribution of real data. GANs consist of two components: a generator and a discriminator. The generator creates synthetic data, while the discriminator's role is to distinguish between real

and generated data. The two networks are trained in a competitive game, where the generator aims to produce data that the discriminator cannot distinguish from real data, and the discriminator strives to correctly identify the authenticity of the data. This adversarial training process leads to the generator improving its ability to produce realistic data over time. GANs have been widely used for tasks such as image generation, data augmentation, and video synthesis.

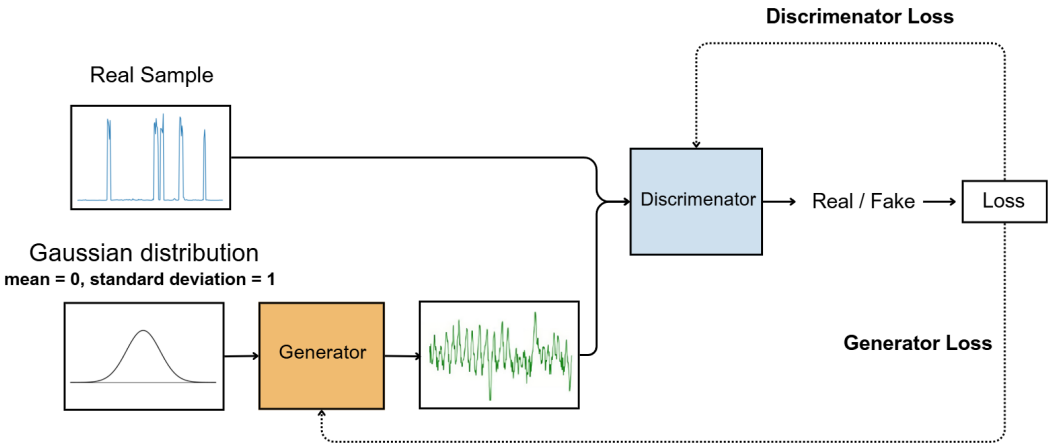

Figure 2.1 – GAN Architecture

2.3.1 GANs Input

The generator takes in random noise sampled from a Gaussian distribution with a mean of 0 and a standard deviation of 1. These choices serve several purposes. Firstly, the mean of 0 ensures that the generated data is centered around zero, preventing any bias in the generation process. This neutral starting point allows the generator to explore both positive and negative variations in the data without any initial skew. Secondly, the standard deviation of 1 provides a consistent level of variability, allowing the generator to create diverse synthetic samples with a wide range of values. This controlled randomness helps the model capture a broad spectrum of energy consumption patterns, promoting better generalization and robustness, particularly in cases where real data is limited or highly varied. Overall, this choice facilitates the generation of diverse yet balanced data, which is crucial for training an effective load disaggregation model.

2.3.2 Generator

The generator in this GAN architecture transforms random input, sampled from a Gaussian distribution, into realistic output through a multi-step process. Initially, as shown in **figure 2.2**, the random noise passes through three **Conv1D blocks,** each designed to capture important data patterns. The first block expands the feature dimensions from **128 to 256 channels**, incorporating **LeakyReLU, BatchNorm1d, Dropout (p=0.3), and Self-Attention** to stabilize training and capture long-range dependencies. The second block reduces the dimensions from 256 to 128 channels, while maintaining data quality with LeakyReLU, BatchNorm1d, and Self-Attention. The third block further refines the features using LeakyReLU and BatchNorm1d. Finally, a Fully Connected Layer scales the output to the required size, incorporating LeakyReLU and BatchNorm1d for smooth learning. This architecture leverages Self-Attention to capture

complex relationships in sequential data, Dropout for regularization, and LeakyReLU to prevent dying neurons. The design ensures effective learning of intricate data patterns, stability during training, and the generation of high-quality synthetic data, making it well-suited for sequential data tasks.

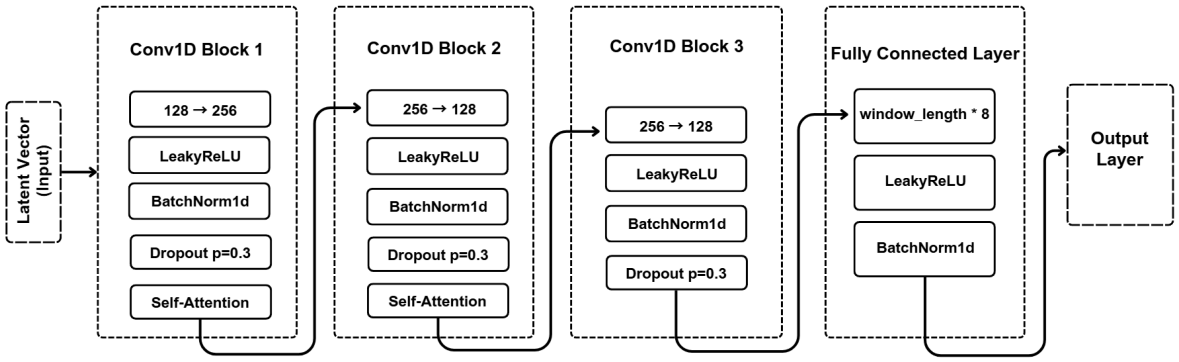

Figure 2.2 – Generator Architecture

The generator's objective is to minimize the loss function, which aims to fool the discriminator into classifying the generated data as real:

$$\min_G L_{G_{ada}} = \mathbb{E}_{g,g'}(k_D(g,g')) + \mathbb{E}_{o,o'}(k_D(o,o')) - 2\mathbb{E}_{o,g}(k_D(o,g)) \tag{2.1}$$

The generator aims to minimize its loss function $L_{G_{ada}}$ by reducing the dissimilarity between the generated samples and real samples, while maximizing the similarity within each class.

2.3.3 Discriminator

The discriminator architecture is designed to distinguish real data from generated data. As shown in **figure 2.3**, it consists of two Conv1D Blocks, where the first block expands the input features from 1 to 32 channels, and the second further increases the channels from 32 to 64. Both blocks share the same components: LeakyReLU activation, Dropout (p=0.3) for regularization, and a Self-Attention mechanism to capture long-range dependencies. After the convolutional layers, the features are flattened into a vector using a Flatten Layer, followed by Fully Connected Layers that apply LeakyReLU and end with a Sigmoid output layer for binary classification ("REAL" or "FAKE"). This architecture employs LeakyReLU to prevent dying neurons, Dropout for regularization, and Self-Attention to enhance learning by focusing on important dependencies, ensuring effective classification of real and fake data.

The discriminator aims to maximize its loss function $\mathcal{L}_D$ to correctly classify real data as real and generated data as fake. To achieve this, it seeks to force generated samples away from the original real data. At the same time, the discriminator minimizes the intra-class variance by implementing a strategy that both reduces the distance between real data points and between generated data points, while maximizing the distance between real and generated samples.

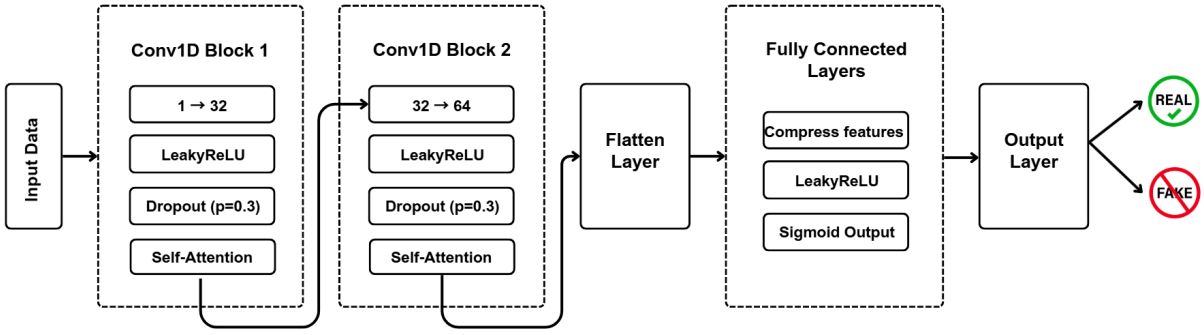

Figure 2.3 – Discriminator Architecture

$$\max_{D} L_{D_{ada}} = \mathbb{E}_{g,g'}(k_D(g, g')) + \mathbb{E}_{o,o'}(k_D(o, o')) - 2\mathbb{E}_{o,g}(k_D(o, g)) \tag{2.2}$$

Here, the term $\mathbb{E}_{g,g'}(k_D(g, g'))$ measures the similarity between two generated samples, $\mathbb{E}_{o,o'}(k_D(o, o'))$ measures the similarity between two real samples, and $\mathbb{E}_{o,g}(k_D(o, g))$ represents the dissimilarity between real and generated samples. The discriminator's objective is to push the generated samples away from real ones while minimizing the variation within each class.

2.3.4 Self-Attention Mechanism

As shown in **figure 2.4** the self-attention mechanism allows the model to focus on different parts of the input sequence by computing a weighted sum of all the elements in the sequence. This is particularly useful for capturing long-range dependencies and intricate relationships between elements in sequential data, which may be overlooked by traditional convolutional or recurrent layers. In the self-attention module, each input element is transformed into three vectors: query, key, and value. The attention scores are computed by taking the dot product of the query and key, followed by a softmax operation to obtain a distribution over the sequence. The final attention output is a weighted sum of the value vectors, where the weights are determined by the attention scores.

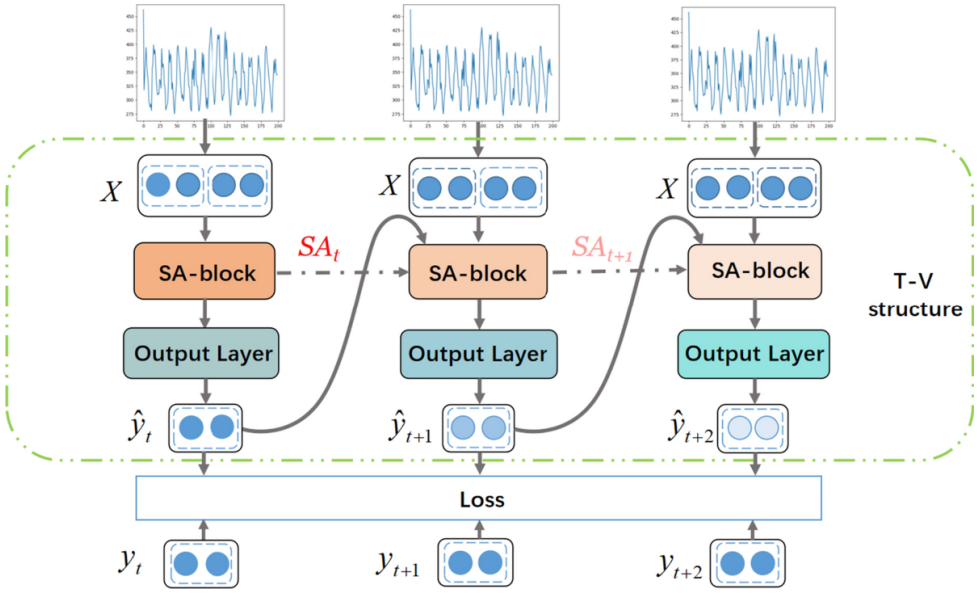

Figure 2.4 – Multi-head Self-Attention Mechanism

The self-attention mechanism is used in both the generator and discriminator of the GAN to improve the model's ability to capture complex relationships within the data. In the generator, self-attention helps produce more coherent and high-quality synthetic data by enabling the network to focus on important features of the generated samples. For the discriminator, self-attention enhances its ability to distinguish between real and fake data by capturing fine-grained details in the input data, leading to better classification performance. By using self-attention, both networks can better model dependencies across different parts of the input, improving the overall quality of the generated data and the robustness of the adversarial training process.

Conclusion

This chapter introduced different key concepts which are crucial for our project. The next chapter will focus on the implementation and results.

Chapter 3

Implementation and Results

Introduction

This chapter focuses on the implementation of the GAN-based approach for generating synthetic power consumption data and its integration into the existing dataset for load disaggregation tasks. Additionally, the evaluation metrics and performance comparisons of the disaggregation model, before and after data augmentation, are presented. The goal is to demonstrate the effectiveness of the synthetic data in improving the model's prediction accuracy and reliability.

3.1 Data Analysis and Preprocessing

3.1.1 Dataset Overview

The REDD is a widely used dataset in the field of energy disaggregation, which involves breaking down aggregate energy consumption into individual appliance-level data. As shown in Table 3.1, the dataset consists of data collected from six different houses, with each house's data organized into separate files called channels. The first two channels record the mains consumption , while the other channels capture the energy consumption of individual appliances, such as refrigerators, microwaves, and ovens. Each channel contains two columns: a timestamp and the corresponding energy consumption value. The mains data is sampled at a rate of one measurement per second, while appliance-level data is sampled every three seconds.

Component	**Description**	**Sampling Rate**	**Data Format**	**Columns**
Mains Consumption	Aggregate power consumption measured the whole-house electricity usage.	sample per second	Time-series (channel_1.dat, channel_2.dat)	Consumption & Timestamp
Appliance Data	Individual power consumption for specific appliances (e.g., oven, fridge, microwave).	sample every 3 seconds	Time-series (channel_3.dat, channel_4.dat, etc.)	Consumption & Timestamp

Table 3.1 – REDD Dataset Specifications

3.1.2 Data Synchronization and Aggregation

To facilitate the manipulation and analysis of the REDD dataset, we centralized all the data into a single CSV file. This step was necessary to make the data more accessible and easier to handle for our analysis. The dataset includes power consumption data for both the mains (whole-house electricity usage) and individual appliances. The mains data was recorded every second, whereas the appliance data was recorded every three seconds. To make the data compatible, we aggregated the mains consumption data to match the sampling rate of the appliance data. This process involved combining the consumption values from the two mains channels to calculate the total energy usage for each house. Once the aggregation was complete, we created a unified CSV file that contains, for each timestamp, the total energy consumption for the house along with the consumption of each individual appliance. This format allows for more effective analysis, enabling us to compare the total household consumption with the usage of individual devices over time.

3.1.3 Data Preprocessing

In the load disaggregation task, the goal is to predict the power consumption of individual appliances from the aggregated household power consumption data. Since the data is time series-based, we utilized the sliding window technique for data preprocessing, which is essential for transforming continuous time series data into smaller, manageable segments that can be fed into the model for training.

3.1.3.1 Sliding Window Technique

The sliding window method divides the time series into smaller overlapping segments or windows. Each window contains a fixed number of consecutive data points, and the window slides one step forward at a time, creating new segments for training. This technique allows the model to learn patterns in smaller chunks of data, improving its ability to handle large datasets with sequential dependencies.

- **Window Length**: The length of each segment (or window) is fixed, and in this case, it is set to 100 time steps. This means each training example corresponds to 100 consecutive data points.
- **Stride**: The stride, or the step size between consecutive windows, is set to 1, meaning the window moves one time step at a time along the sequence.

3.1.3.2 Data Cleaning and Normalization

Before segmentation, the data undergoes two important preprocessing steps:

1. **Removal of Abnormal Points**: Abnormal data points that deviate significantly from the surrounding values are identified and removed. This ensures that the model is trained on clean, representative data.
2. **Normalization**: Both the aggregated (input data) and target values (appliance consumption) are normalized. Normalization scales the data to a range between 0 and 1, making it easier for the model to learn and preventing any one feature from dominating due to differences in scale.

3.1.3.3 Segmenting the Data

Once the data is cleaned and normalized, it is segmented into smaller windows using the sliding window technique. For each window:

- A fixed-length segment (100 data points) is extracted.
- The target value for each segment is determined by taking the average value of the target variable (the power consumption of the appliance) within the segment. This average value is used as the target for the model.

3.1.3.4 Training Data Preparation

After segmenting the data, the individual segments are accumulated into a final training dataset. The feature data from each appliance (such as total household consumption or individual appliance readings) and their corresponding target values (the consumption of the appliance) are stacked together to form the complete training set.

The resulting training data consists of:

- **Features**: Segmented windows of the input data (aggregated consumption).
- **Targets**: The corresponding average target values for each window, representing the consumption of the individual appliance.

This data is now ready for use in training the model, which will learn to predict the appliance's consumption from the aggregated data.

3.2 GAN Training Process

The GAN was trained to generate 15-minute windows of microwave power consumption data, using a latent input sampled from a Gaussian distribution with a mean of 0, variance of 1, and a latent dimension of 64. The training required 5000 epochs to converge, and memory limitations were overcome by training on 15-minute windows (300 timesteps), which were later combined to create synthetic daily consumption patterns.

3.2.1 Training Parameters

The table below 3.2 summarizes the training parameters used for both the generator and discriminator:

Component	Generator	Discriminator
Learning Rate	0.00005	0.0003
β_1	0.5	0.5
β_2	0.999	0.999
Epochs	5000	5000

Table 3.2 – GAN Training Hyperparameters

The use of different learning rates for the generator and discriminator in GANs is crucial for maintaining training stability. Since the discriminator typically converges faster due to its simpler task of distinguishing real from fake data, a higher learning rate is used for it. On the other hand, the generator's learning process is more complex, requiring gradual adjustments to produce realistic data. A smaller learning rate for the generator helps prevent it from making large, destabilizing updates, allowing it to learn more effectively from the discriminator's feedback and resulting in better-quality generated data over time.

3.3 Synthetic Data Integration

To augment the dataset with synthetic signals, the following steps were taken:

1. Active consumption windows of the microwave were identified in the original dataset.
2. The original microwave signal was subtracted from the aggregate signal.
3. Synthetic signals generated by the GAN were used to replace the original microwave data.
4. The adjusted aggregate signal was recombined with the synthetic microwave signal, creating a realistic and augmented dataset.

This approach preserved the integrity of the aggregate signal while seamlessly introducing synthetic data.

3.3.1 Synthetic Signal Visualizations

Visual comparisons of real versus synthetic microwave signals indicate that the GAN successfully captured key patterns in the appliance's power consumption. These synthetic signals are visually indistinguishable from real signals, validating the GAN's effectiveness.

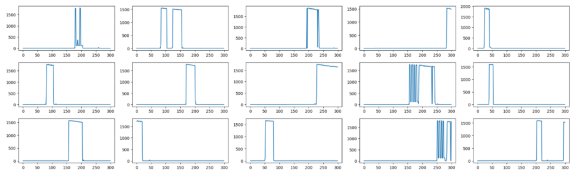

Figure 3.1 – Real Microwave Active Power Consumption Signal samples.

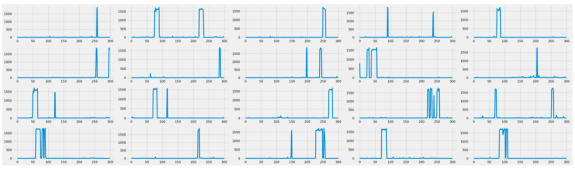

Figure 3.2 – Synthetic Microwave Active Power Consumption Signal samples.

Figure 3.3 – Comparison of Real vs. Synthetic Microwave Active Power Consumption Signals.

3.4 Results

3.4.1 Performance Metrics

To evaluate the effectiveness of the load disaggregation model before and after data augmentation, the following metrics were calculated:

- **Precision:** Measures the proportion of true positives among all predicted positives.
- **Recall:** Measures the proportion of true positives among actual positives.
- **F1-Score:** The harmonic mean of precision and recall.
- **Accuracy:** Measures the overall correctness of the model.
- **R-squared (R^2):** Indicates how well the predicted signal matches the ground truth signal.
- **MAE:** Calculates the average magnitude of errors between predictions and ground truth.
- **MSE:** Evaluates the average squared difference between predicted and actual values.
- **RETE:** Measures the relative difference in total energy consumption between predicted and actual signals.

The metrics were calculated using Python functions, with thresholds applied to classify predictions as "on" or "off" states. The relevant code for these calculations is included in the appendix.

3.4.2 Comparison Before and After Augmentation

3.4.2.0.1 Metric Improvements: After incorporating synthetic data generated by the GAN, the model's performance metrics showed significant improvement. Table 3.4.2.0.1 summarizes these metrics:

Metric	Before Augmentation	After Augmentation
Precision	0.36	0.96
Recall	0.75	0.99
F1-Score	0.48	0.98
Accuracy	0.9	0.99
R-squared (R^2)	0.32	0.98
MAE	15 KW	3.96 KW
MSE	8236	278
RETE	0.43	0.24

Table 3.3 – Performance Metrics Before and After Augmentation

3.4.2.0.2 Loss and Error Plots:

- **Training Loss and MAE:** Plots comparing the disaggregation model's loss and MAE before and after augmentation highlight reduced errors after augmentation.

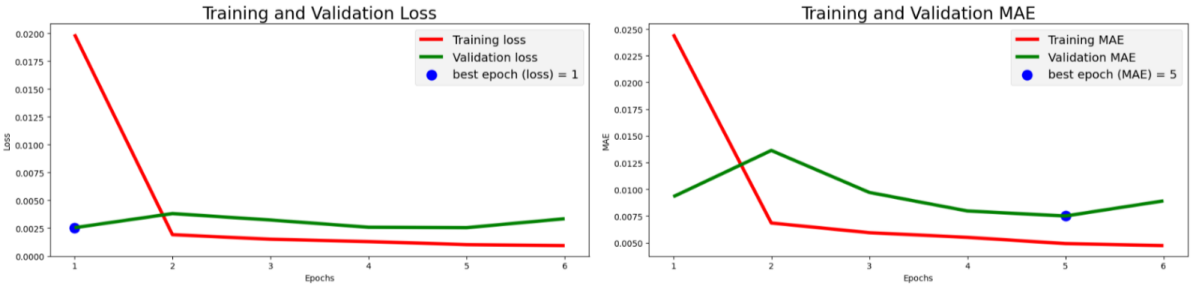

Figure 3.4 – Loss and MAE before Augmentation

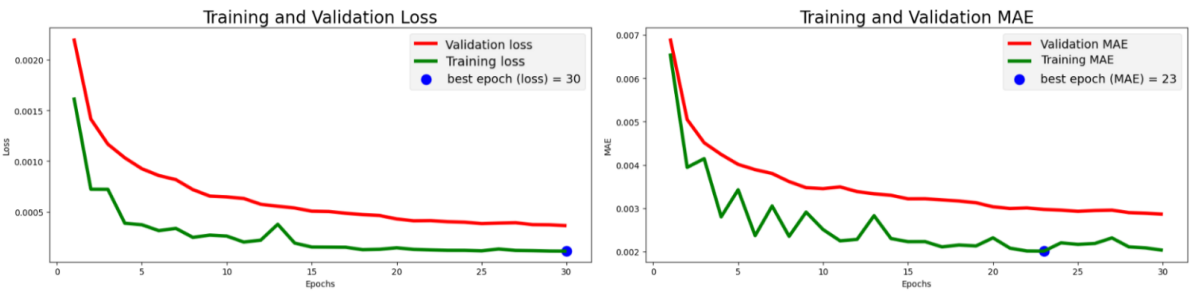

Figure 3.5 – Loss and MAE after Augmentation

- **Prediction Comparisons:** Figures 3.6 show predictions versus ground truth before and after augmentation, demonstrating that the model trained on augmented data better captures the appliance-level consumption patterns.

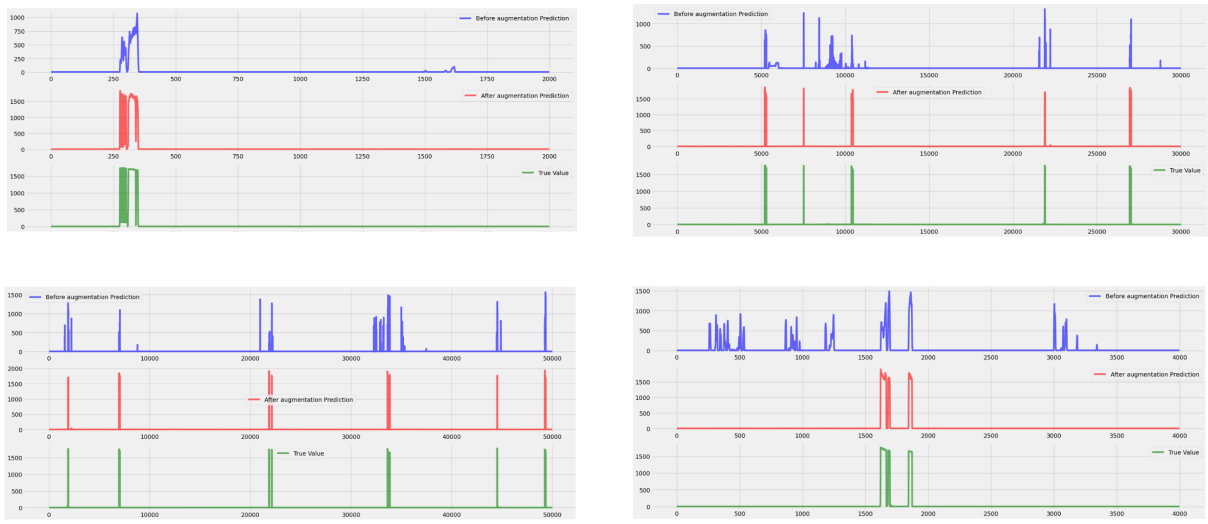

Figure 3.6 – Comparison of Predictions vs Ground Truth before and after augmentation.

3.4.3 Challenges and Limitations

Despite the improvements, some challenges remain:

- Generating longer continuous signals (e.g., full-day consumption) is computationally intensive.
- The current approach assumes stationary appliance behavior, which may not generalize to appliances with highly dynamic patterns.

Conclusion

The implementation of GAN-based synthetic data generation demonstrated its effectiveness in addressing the limitations of an imbalanced dataset for load disaggregation tasks. By integrating realistic synthetic data into the original dataset, the model achieved significant improvements in precision, recall, and overall accuracy. These results underscore the potential of GANs to enhance the performance of energy disaggregation models and pave the way for further research in the field.

Conclusion

This project, titled **Data Augmentation Strategy Using GAN for Load Disaggregation** has been a significant undertaking, developed over a dedicated period. The primary aim of this project was to implement a data augmentation approach using GANs to address challenges in load disaggregation for more efficient energy management. Throughout this journey, we have explored various tools and methodologies, such as GANs, to enhance the model's ability to predict energy consumption patterns, particularly for less frequently used appliances.

This report has provided a comprehensive overview of the tools used, the concepts applied, and the iterative approaches taken to refine the model. These efforts have ensured that our methodology aligns with the project's goals, ultimately improving the accuracy and efficiency of load disaggregation.

Looking ahead, we acknowledge the technical challenges that remain, particularly the need for large, high-quality data to further enhance the model's performance. However, the foundation laid in this project sets the stage for continued innovation and progress in the field of energy management.

Throughout the 4-month duration of this tutored project at **Sagemcom**, we have deepened our understanding of theoretical concepts while gaining valuable practical insights critical for addressing real-world energy management challenges. The collaborative nature of the project and the learning experiences it offered have significantly enriched our knowledge, setting the foundation for future research and advancements in this field.

Bibliography

[1] Sagemcom. **Consulted on 14/01/2025**.

[2] Jack Kelly and William Knottenbelt. Neural NILM: Deep Neural Networks Applied to Energy Disaggregation. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments (BuildSys'15)*, November 2015. **Consulted on 28/09/2024**.

[3] Asma Maalej and Chiheb Rebai. Sensor Data Augmentation Strategy for Load Forecasting in Smart Grid Context. *University of Carthage, SUP'COM, COSIM Research Lab.*, 2021. **Consulted on 29/09/2024**.

[4] Ferdaous Chaabane, Safa Réjichi, and Florence Tupin. Self-Attention Generative Adversarial Networks for Time Series VHR Multispectral Image Generation. *IEEE Explore*, 2021. **Consulted on 14/11/2024**.

[5] Time Series. **Consulted on 14/01/2025**.

[6] Sliding Window. **Consulted on 14/01/2025**.