REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre :Startup_RTIC 4/RTIC/M2/2023

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Réseaux et Technologies de l'Information et de la Communication (RTIC)**

AI-based Energy Management System for Smart Grid Network

Par :

SLATNIA NAIMA ET TOUAREF NOURELHOUDA

Soutenu le 03/07/2023 devant le jury composé de :

Guemeida Abdelbasset	MCB	Président
Naidji Ilyes	MCB	Rapporteur
Sahli Sihem	MAA	Examinateur

Année universitaire 2022-2023

Declaration of authorship

I hereby declare that the report submitted is my own unaided work. All direct or indirect sources used are acknowledged as references. I am aware that the Thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the report as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future Theses submitted. Further rights of reproduction and usage, however, are not granted here. This paper was not previously presented to another examination board and has not been published.

Abstract

The development of the traditional electrical grid into the smart grid has lead to need for modern energy management systems that takes into consideration the variability of renewable energy and electrical loads that have an intermittent nature and need to be forecasted. For this reason, artificial intelligence can play a pivotal role in the control and monitoring of the smart grid.

This project aims to provide an AI-based solution for smart grid in order to improve the energy cost and electricity consumption price. To achieve this aims, we propose to develop a load forecasting method at the distribution level in order to reduce energy cost. We propose also to develop an electricity price forecasting method at the consumption level in order to reduce electricity consumption cost. During this study, the LSTM model is selected as the best model for performing time-series forecasting.

In order to show the accuracy of the proposed forecasting models, a comparative study is provided between the developed model (LSTM) and other existing methods such as FNN and GRU. Furthermore, In order to demonstrate the e↵ectiveness of the proposed solution, simulation tests of smart grid network is carried which shows a significant gain in terms of energy cost and electricity price.

Résumé

Le développement du réseau électrique traditionnel vers le réseau électrique intelligent a conduit à la nécessité de systèmes modernes de gestion de l'énergie qui prennent en considération la variabilité des énergies renouvelables et des charges électriques qui ont un caractère intermittent et doivent être prévues. Pour cette raison, l'intelligence artificielle peut jouer un rôle central dans le contrôle et la surveillance du réseau électrique intelligent.

Ce projet vise à fournir une solution basée sur l'IA pour les réseaux intelligents afin d'améliorer le coût de l'énergie et le prix de la consommation d'électricité. Pour atteindre cet objectif, nous proposons de développer une méthode de prévision de charge au niveau de la distribution afin de réduire les coûts énergétiques. Nous proposons également de développer une méthode de prévision du prix de l'électricité au niveau de la consommation afin de réduire le coût de la consommation d'électricité. Au cours de cette étude, le modèle LSTM est sélectionné comme le meilleur modèle pour effectuer des prévisions de séries chronologiques.

Afin de montrer la précision des modèles de prévision proposés, une étude comparative est fournie entre le modèle développé (LSTM) et d'autres méthodes existantes telles que FNN et GRU. De plus, afin de démontrer l'efficacité de la solution proposée, des tests de simulation de réseau intelligent sont effectués, ce qui montre un gain significatif en termes de coût de l'énergie et de prix de l'électricité.

Dedication

I dedicate this dissertation work to My dear father , my precious and beloved mother , I don't forget my dear brothers and sisters and of course the rest of the family and friends. and thanks to my supervisor sir ILYES NAIDJI for his advices and his help.

Touaref Nourelhouda

Acknowledgements

I would like to express my heartfelt gratitude to my parents for their unwavering support and guidance throughout my academic journey. They have been my pillars of strength, always believing in me and encouraging me to reach for my dreams,to my sisters and my brothers.

I'm also deeply grateful to my supervisor ILYES NAIDJI for his guidance and mentorship that have shaped my understanding and nurtured my growth. His patience, expertise and dedication have played a vital role in shaping this project. I'm thankful for his invaluable guidance that allowed me to explore and develop in my research endeavors.

Finally, I would also like to express my gratitude to my friends who pushed me to achieve this work, especially Chourouk, Youssra, Romaissa and all my classmates, and I am very grateful to all of you.

Selatnia Naima

Contents

General introduction

A smart grid system refers to an advanced electrical grid that incorporates modern communication, computing, and control technologies to enhance the efficiency, reliability, and sustainability of electricity generation, transmission, distribution, and consumption. It is an evolution of the traditional electrical grid that enables two-way flow of information and power between utilities and consumers.
The primary goal of a smart grid system is to optimize the operation of the entire electricity ecosystem, from power plants to end-users, by leveraging real-time data, automation, and intelligent decision-making. Here are some key components and features of smart grid systems:

- Advanced Metering Infrastructure (AMI): Smart grid systems often involve the deployment of smart meters, which provide detailed information about electricity consumption at individual homes or businesses. These meters enable two-way communication, allowing utilities to gather accurate real-time data and consumers to monitor and manage their energy usage more effectively.

- Distributed Energy Resources (DERs) Integration: Smart grid systems accommodate the integration of diverse distributed energy resources, such as solar panels, wind turbines, energy storage systems, and electric vehicles. These resources can be seamlessly connected to the grid and managed in a coordinated manner, allowing for better utilization of renewable energy and improving grid stability.

- Demand Response and Load Management: Smart grids facilitate demand response programs that encourage consumers to adjust their electricity usage during periods of high demand or supply constraints. Through pricing signals or automated control systems, consumers can reduce or shift their electricity consumption, helping to balance the grid and optimize its utilization.

- Energy Management and Optimization: Smart grid systems enable advanced energy management techniques, utilizing data analytics, machine learning, and optimization algorithms. These tools help utilities and consumers make informed decisions about energy production, consumption, and storage, leading to improved efficiency and cost savings.

- Cybersecurity and Resilience: With increased connectivity and data exchange, smart grid systems emphasize robust cybersecurity measures to protect against potential cyber threats and unauthorized access. Ensuring the resilience and reliability of the grid infrastructure is of utmost importance to maintain a secure energy supply.

The deployment of smart grid systems offers numerous benefits, including reduced energy waste, increased integration of renewable energy sources, improved outage management, lower operational costs, and enhanced consumer engagement in energy management. By transforming the traditional electrical grid into an intelligent and interactive network, smart grids pave the way for a more sustainable, efficient, and reliable electricity future.

Problem statement

The main problematic of this project is to improve energy management of smart grid at distribution level and consumption level. Due to the intermittent nature of smart grid that is based on renewable energy and electrical loads, a prediction method should be used.

- A load forecasting method should be developed in order to predict electrical loads in order to reduce energy cost.
- An electricity price forecasting method should be developed in order to predict electricity price in order to reduce electricity consumption cost.

Contributions

The contributions of this project are:

- A load forecasting method is developed based on LSTM model which is advanced form of recurent neural networks.
- An electricity price forecasting method is developed which is based on LSTM model that is the best for tim-series forecasting.
- Simulation tests are carried on a real smart grid network in order to demonstrate the effectiveness of the proposed AI-based solution for energy management in smart grid.

Thesis structure

This thesis is organized as follows:

- Chapter 1 introduces the concepts of smart grid and energy management. In addition, chapter 1 defines the key components of the smart grid.
- Chapter 2 provide an introduction about the artificial inteligence and its application in smart grid. Furthermore, chapter 2 reviews some related work on artificial intelligence in smart grid.
- Chapter 3 designs our model for improving energy management in smart grid. Chapter 2 provide mathematical formulation of smart grid in both distribution and consumption level and shows the developed LSTM model for load and electricty price forecasting.
- Chapter 4 provide the code for developing our AI-based solution and the related results. It provide also simulation results of smart grid after working with the proposed LSTM model.

1 Smart Grid concepts

Introduction

A smart grid is a modernized electrical grid that uses digital communication technology to improve the efficiency, reliability, and sustainability of electricity generation, transmission, and distribution. It is an advanced system that integrates renewable energy sources, energy storage, and advanced metering infrastructure to provide better control and management of the electricity network (1) .
Smart grids enable two-way communication between electricity producers and consumers, allowing for real-time monitoring and optimization of energy usage. This results in increased energy efficiency, reduced power outages, and improved system reliability (2).
Moreover, smart grids can also support the integration of electric vehicles, which can be used as a distributed energy resource to provide grid services such as frequency regulation, load balancing, and peak shaving (3).
Smart grids also offer numerous benefits for utilities, such as reducing operational costs, improving asset management, and enabling predictive maintenance (4).
Smart grids have gained widespread attention and support from governments, utilities, and industry stakeholders worldwide. Many countries have already initiated smart grid programs and projects to accelerate the deployment of this technology (5).
So in this chapter, we will discuss concepts related to the smart grid, its components, and how energy is managed.

Smart Grid Composition

A smart grid is a modernized electrical grid that uses advanced technologies to efficiently manage and optimize the generation, distribution, and consumption of electricity. The composition of a smart grid includes several key components, including:

bus

In a smart grid, a bus is a node that connects different electrical components, such as transformers, generators, and loads, to form a power network. It acts as a central point where multiple electrical connections converge and distribute power to different parts of the grid (6).

A node, on the other hand, is a point in the smart grid network where different electrical components are connected. It can be a substation, distribution automation device, advanced meter, energy storage system, or home energy management system. Nodes can communicate with each other and exchange information to optimize the performance of the grid (7).

Transmission lines

Transmission lines are a critical component of the smart grid infrastructure. They are the high-voltage power lines that carry electricity over long distances from power plants to distribution substations, where the voltage is stepped down for distribution to homes and businesses (8).

In a smart grid, transmission lines are equipped with sensors and monitoring devices that enable real-time monitoring of the flow of electricity and the health of the transmission infrastructure. This data is transmitted to a central control system, where it is analyzed to optimize the flow of electricity, detect and isolate faults, and manage congestion on the grid (9).

Overall, transmission lines play a crucial role in the composition of the smart grid, enabling the reliable and efficient delivery of electricity to consumers while facilitating the integration of renewable energy sources and other distributed energy resources.

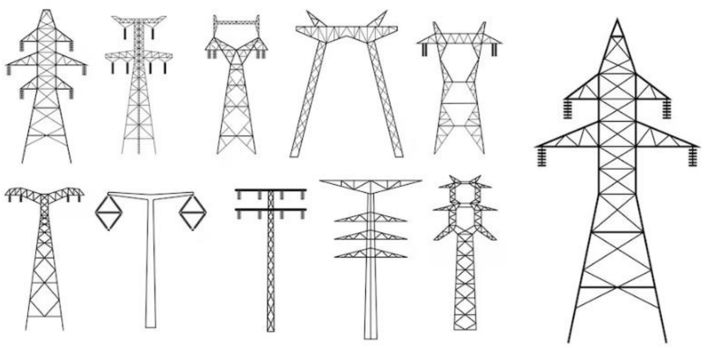


Figure 1: Examples of a Transmission lines

Generator

A generator is a key component of the smart grid infrastructure, responsible for producing electricity that can be distributed to homes and businesses.

In the traditional power grid, most electricity is generated at centralized power plants and transmitted over long distances to distribution substations before being distributed to end-users. In a smart grid, however, generators can include a variety of distributed energy resources (DERs) that are located closer to where the electricity is needed, such as solar panels, wind turbines, and small-scale generators (10).

The smart grid includes technologies that enable better control and integration of these DERs into the overall grid, making it possible to more e↵ectively manage the variability and intermittency of renewable energy sources. For example, advanced control systems can enable generators to respond quickly to changes in energy demand and supply, while energy storage systems can help balance the supply and demand of electricity (11).

Overall, generators play a crucial role in the composition of the smart grid, helping to provide a more reliable, sustainable, and cost-e↵ective supply of electricity to consumers.

Figure 2: Examples of a Generator

Load

Load is another important component of the smart grid infrastructure. It refers to the amount of electricity that is consumed by homes, businesses, and other end-users connected to the grid.

In a traditional power grid, load is typically managed through the use of centralized power plants that generate electricity at a steady rate to meet the expected demand. However, in a smart grid, load management is more flexible and dynamic, thanks to the integration of advanced sensors, control systems, and communication technologies.

With these technologies, the smart grid can better monitor and respond to changes in load, adjusting the supply of electricity to match the demand in real-time. For example, smart meters can provide detailed information on energy consumption patterns, allowing consumers and utilities to identify opportunities for energy savings and more efficient use of electricity.

Load management is also important for integrating renewable energy sources, which are often intermittent and variable in their output. By better matching the supply and demand of electricity, load management technologies can help ensure that renewable energy sources are used e↵ectively and efficiently (12).

Overall, load management is a crucial part of the composition of the smart grid, helping to ensure the reliability, efficiency, and sustainability of the electricity supply.

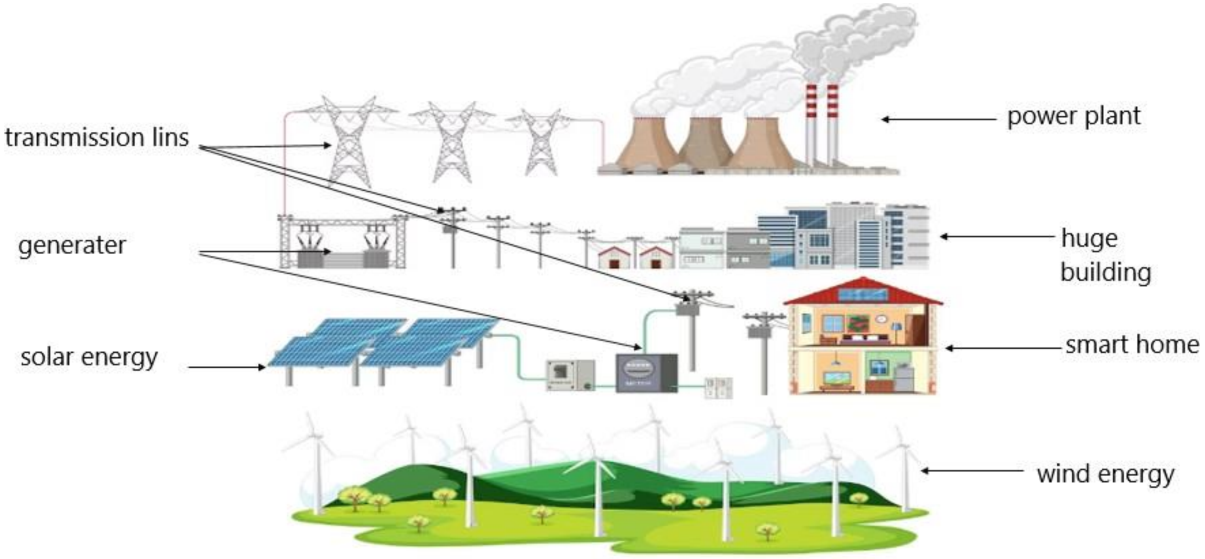


Figure 3: Smart Grid Composition

Smart Grid Levels

Smart grids typically have four levels:

Generation level

The generation level of the smart grid represents the initial phase of the electricity supply chain, wherein power is generated using diverse energy sources like coal, natural gas, nuclear, hydro, wind, solar, and other renewables. Its fundamental objective is to produce electricity in an efficient, cost-e↵ective, and sustainable manner. The integration of smart grid technologies holds immense potential for enhancing the generation level by facilitating the incorporation of renewable energy sources and improving the overall efficiency of conventional power plants.

Advanced sensors and control systems stand as prime examples of technologies that can bolster the efficiency of power plants. By optimizing fuel consumption, reducing emissions, and minimizing downtime, these intelligent systems enable power plants to operate more efficiently. Through real-time monitoring and precise data analysis, smart grid technologies empower operators to make informed decisions, thereby maximizing the efficiency of the generation process. This not only leads to cost savings but also contributes to environmental sustainability by reducing carbon emissions.

Moreover, smart grid technologies play a crucial role in achieving a balance between supply and demand. Demand response programs are a prime illustration of how the smart grid can achieve this equilibrium. These programs encourage consumers to curtail their electricity usage during periods of peak demand. By utilizing price signals and incentives, consumers can adjust their energy consumption habits, effectively reducing strain on the grid. Consequently, this mitigates the need for costly infrastructure expansion and promotes a more sustainable and reliable energy supply. Additionally, the integration

Figure 4: Generation level

of renewable energy sources is a key focus of the generation level in the smart grid. The variability and intermittency of renewable sources like wind and solar power pose challenges to grid stability. However, smart grid technologies offer solutions to address these challenges. Advanced forecasting models, coupled with real-time monitoring and control systems, enable grid operators to efficiently integrate renewable energy sources into the generation mix. By optimizing the utilization of renewable sources, the smart grid promotes the transition to cleaner and more sustainable energy systems.
In summary, the generation level of the smart grid is the initial stage of the electricity supply chain, encompassing the production of electricity from diverse energy sources. The integration of smart grid technologies enables the efficient integration of renewable energy sources, enhances the efficiency of traditional power plants, and facilitates a balanced supply-demand relationship. By maximizing efficiency, reducing emissions, and promoting the use of clean energy sources, the generation level of the smart grid contributes to a more sustainable and resilient energy infrastructure.

Transmission level

The Transmission level stands as the second tier within the framework of the smart grid. This level plays a vital role in the efficient and reliable transportation of high-voltage electricity over long distances, spanning from the generation level to the distribution level. It encompasses a network of high-voltage transmission lines, transformers, and various equipment explicitly designed to transmit substantial amounts of electricity across extensive distances.

The integration of smart grid technologies holds the potential to enhance the performance of the transmission level by enabling real-time monitoring and control of the transmission network. Phasor measurement units (PMUs) serve as a notable example of such technologies. These units provide high-resolution data on the status and condition of the transmission network, equipping utilities with valuable insights to swiftly detect and respond to potential issues. By obtaining accurate and up-to-date information, utilities can proactively address any anomalies or disturbances, thereby minimizing downtime and optimizing the reliability of the transmission infrastructure.

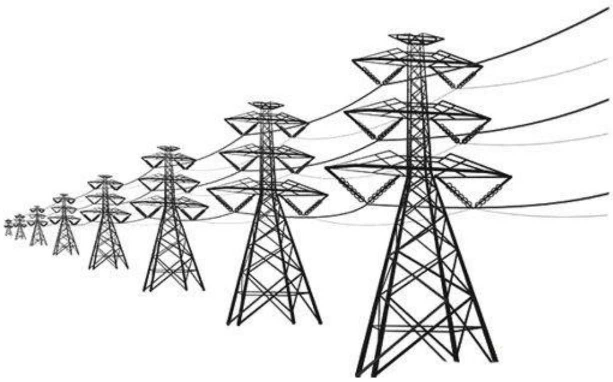


Figure 5: Transmission level

Moreover, advanced transmission control systems play a pivotal role in optimizing the utilization of the transmission network. By facilitating more efficient and reliable power flow, these systems contribute to maximizing the capacity and effectiveness of the transmission infrastructure. Furthermore, the integration of renewable energy sources is another aspect that can be improved at the transmission level. The intermittent nature of renewable sources, such as wind and solar power, can be effectively managed through smart grid technologies. By enabling the seamless integration of these intermittent sources, the transmission level supports a more robust and sustainable energy mix.
Furthermore, the transmission level of the smart grid can serve as a catalyst for the development of new transmission infrastructure. An example of such infrastructure is high-voltage direct current (HVDC) lines. These advanced transmission lines offer enhanced efficiency and lower losses compared to traditional alternating current (AC) lines. HVDC lines have the capability to transmit renewable energy generated in remote areas to densely populated urban centers, enabling the efficient long-distance transportation of clean energy. This infrastructure expansion is essential for unlocking the full potential of renewable resources and supporting the transition to a greener and more decentralized energy system.
In summary, the Transmission level within the smart grid architecture plays a critical role in transporting high-voltage electricity over long distances. The integration of smart grid technologies enhances this level by enabling real-time monitoring and control, optimizing power flow, improving the integration of renewable energy sources, and facilitating the development of new transmission infrastructure. By leveraging these advancements, the transmission level contributes to a more resilient, efficient, and sustainable electricity transmission network.

Distribution level

The third level where the electricity is generated at the power plants and is transmitted to the substations. The substations are responsible for controlling and monitoring the distribution of power to individual consumers. Smart grid technologies can improve the distribution level by enabling real-time monitoring and control of the distribution network. For example, advanced sensors and smart meters can provide real-time data on energy usage and grid conditions, which can help utilities detect and respond to outages and other issues more quickly. Moreover, advanced distribution automation systems can

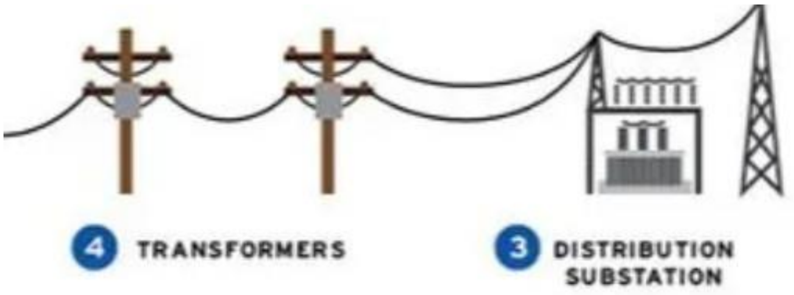


Figure 6: Distribution level

help optimize the distribution network, by enabling automatic switching of power lines, voltage regulation, and load balancing. These systems can also improve the integration of renewable energy sources, by enabling the smooth integration of intermittent sources such as solar and wind.

Finally, the distribution level of the smart grid can also facilitate the integration of distributed energy resources (DERs) such as rooftop solar panels, energy storage systems, and electric vehicles. By enabling two-way communication and control between these resources and the grid, utilities can better manage the supply and demand of electricity and maximize the use of renewable energy sources.

Consumption level

The consumer level of the smart grid represents the portion of the electrical grid that directly involves end-users, encompassing both households and businesses. This facet of the grid is commonly referred to as the ”smart home” or ”smart building” concept, with the primary objective of granting consumers greater control over their energy consumption and expenses. At the heart of the consumer level lie smart meters, advanced digital devices employed to measure and monitor the electricity usage of consumers. These cutting-edge meters provide real-time information regarding energy consumption, equipping consumers with valuable insights and enabling them to make informed decisions about their energy usage. Armed with precise and up-to-date data, consumers can identify energy-intensive activities and modify their behavior accordingly, optimizing efficiency and reducing waste. Furthermore, smart meters facilitate remote management of electricity flow to consumers' premises, allowing for features such as on-demand disconnection and reconnection. These capabilities e↵ectively curb unnecessary energy usage, resulting in cost savings.

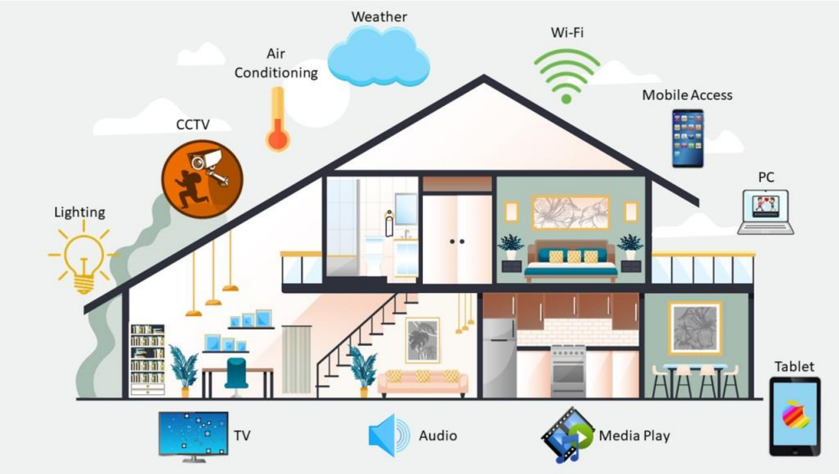


Figure 7: Consumption level

In addition to smart meters, the consumer level of the smart grid incorporates a diverse range of intelligent devices and appliances. For instance, smart thermostats empower users to control and schedule the temperature settings of their heating and cooling systems, promoting energy efficiency. Smart lighting systems o⅃er automated lighting control and energy-efficient options, optimizing electricity usage in the realm of illumination. Moreover, various smart appliances, such as refrigerators, washing machines, and dishwashers, can be seamlessly integrated into the smart grid ecosystem. This integration allows users to remotely control and schedule the operation of these appliances, thereby enabling efficient energy usage.
Communication technologies serve as the bridge between these devices and appliances at the consumer level and the broader smart grid. Through smartphone apps or other digital interfaces, consumers gain the ability to monitor and manage their energy consumption from any location. This newfound capability empowers users to adapt their energy usage based on personal preferences, time-of-use pricing, and the overall energy demand within the grid. As a result, consumers can reduce costs and lessen their environmental impact by optimizing energy usage. In conclusion, the consumer level of the smart grid provides numerous benefits to end-users. Through the utilization of smart meters, interconnected devices, and appliances, consumers are empowered to make informed decisions, optimize their energy usage, and actively contribute to a more sustainable and efficient electrical grid. By fostering greater control and awareness, the consumer level of the smart grid paves the way for a more energy-conscious society.

Smart Grid Global Architecture

There are two types of architecture for a smart network :
the first one electrical architecture :

The architecture of modern smart grids shifted from the centralised one to one that is highly distributed. After the appearance of distributed energy sources, the smart grid network becomes more and more autonomous and self-healing, thanks to the improvement of energy availability and reliability.

The figure shows the global architecture of the smart grid network which consists of four level that are generation, transmission, distribution and consumption. Electricity generation, transmission and distribution

The power generation begins at generation parks With huge power plants such as nuclear, thermal or hydro power plants.

The power is transferred then in high voltage to transmission consumers such as large factories. The power is then transformed in medium voltage in transmission substations and transferred to sub-transmission consumers such as huge buildings.

The power then is transformed in Low voltage in distribution substations and transferred to Low voltage consumers such as smart homes.

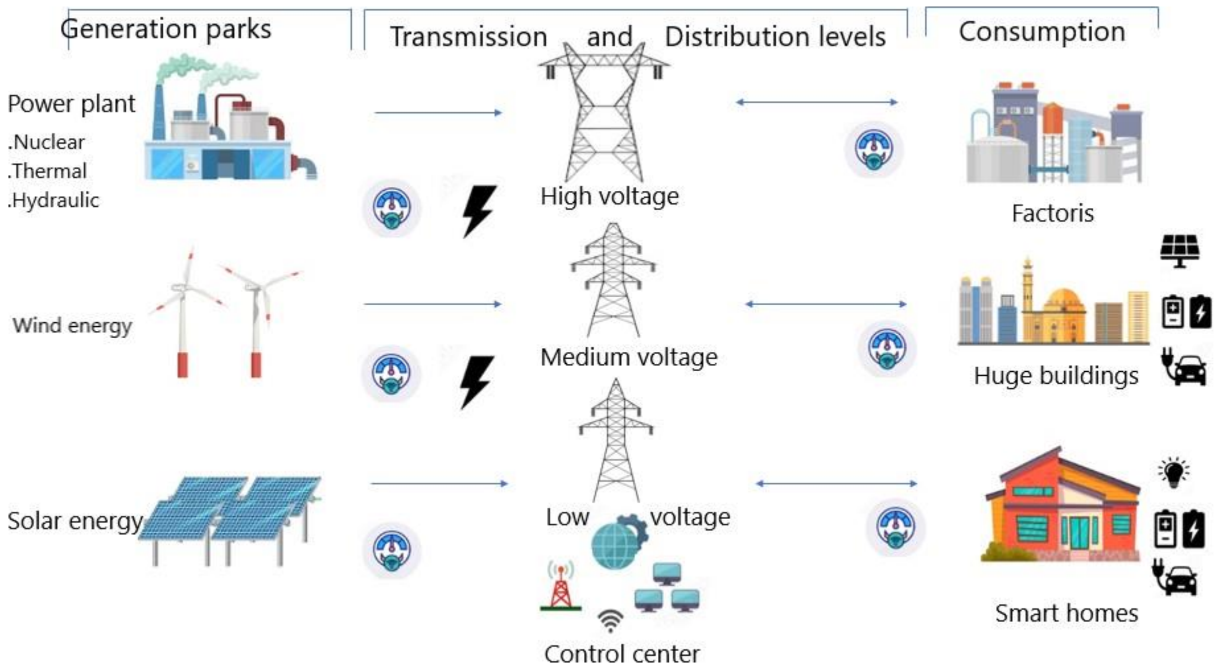


Figure 8: Smart Grid global architecture

and the second one physical architecture :

The Smart Grid Global Architecture is a conceptual model that describes the di↵erent components and interactions of a smart grid. It is divided into three main layers: the layer, the communication layer, and the application layer.

The physical layer is responsible for the transmission and distribution of electricity. It includes the power generation facilities, the transmission lines, and the distribution lines.

The communication layer is responsible for the exchange of information between the different components of the smart grid. It includes the smart meters, the sensors, and the communication networks.

The application layer is responsible for the use of the information that is exchanged in the communication layer. It includes the applications that are used to manage the grid, to control the demand for electricity, and to provide new services to customers.

The Smart Grid Global Architecture is a complex system that is still under development. However, it has the potential to revolutionize the way that electricity is generated, transmitted, and distributed.

Here are some of the benefits of the Smart Grid Global Architecture:

Increased efficiency: The Smart Grid Global Architecture can help to improve the efficiency of the power grid by reducing losses and improving the reliability of the grid.

Reduced costs: The Smart Grid Global Architecture can help to reduce the costs of electricity by making the grid more efficient and by providing new services to customers.

Increased security: The Smart Grid Global Architecture can help to improve the security of the power grid by making it more difficult for hackers to disrupt the grid.

Increased sustainability: The Smart Grid Global Architecture can help to increase the sustainability of the power grid by making it easier to integrate renewable energy sources into the grid.

The Smart Grid Global Architecture is a promising technology that has the potential to improve the efficiency, reliability, security, and sustainability of the power grid.

Control and monitoring

Control and monitoring are critical components of the smart grid infrastructure, enabling real-time management of the flow of electricity and the health of the grid.

Control systems use advanced algorithms and models to optimize the operation of the grid, taking into account factors such as energy demand, the availability of renewable energy sources, and the condition of transmission and distribution infrastructure. These systems can automatically adjust the supply and demand of electricity to maintain a stable and reliable grid, and they can also detect and isolate faults or other problems before they lead to outages or other disruptions.

Monitoring systems use a variety of sensors and communication technologies to provide real-time data on the performance of the grid, including information on energy flows, voltage levels, and the health of individual components such as transformers or circuit breakers. This data is analyzed by control systems and used to inform decisions about the operation of the grid, such as which generators to dispatch or which lines to de-energize in the event of an outage.

Together, control and monitoring systems enable a more flexible, dynamic, and responsive grid that can adapt to changes in energy demand and supply, while also providing greater visibility and control over the health of the infrastructure. This helps to ensure the reliability, efficiency, and resilience of the electricity supply, while also enabling the integration of new technologies and renewable energy sources.

Energy Management

Energy management for a smart grid involves the implementation of strategies and technologies that enable efficient and reliable distribution, monitoring, and utilization of energy. Here are some key aspects of energy management for a smart grid:

1-Demand Response: A smart grid should be able to balance energy demand and supply by managing consumption during peak periods. Demand response is a key strategy to achieve this balance, which involves incentivizing consumers to adjust their energy consumption during peak periods.

2-Energy Storage: Energy storage systems like batteries can help to store excess energy produced during low-demand periods and release it during peak demand periods. This can help to balance the supply and demand of energy and increase the reliability of the grid.

3-Distributed Energy Resources (DERs): DERs such as solar panels, wind turbines, and microgrids can be integrated into a smart grid to increase the efficiency of energy distribution and utilization. They can also help to reduce the dependence on centralized power plants.

4-Advanced Metering Infrastructure (AMI): AMI allows for the collection of real-time data on energy consumption, which can help to optimize energy management strategies and improve the efficiency of the grid.

5-Artificial Intelligence (AI): AI can be used to analyze energy consumption data and predict future demand, which can help to optimize energy management and reduce costs.

Energy Management Concepts

Energy Audit: An energy audit involves a comprehensive assessment of energy consumption patterns, identifying areas of energy waste, and suggesting measures for improvement. It helps in understanding the energy usage and finding opportunities for energy conservation.

Energy Efficiency: Energy efficiency focuses on reducing energy consumption while maintaining the same level of output or service. This involves implementing measures like upgrading to energy-efficient appliances, optimizing system operations, using insulation, and adopting energy-efficient lighting solutions.

Demand-Side Management (DSM): DSM involves strategies and techniques to influence consumer energy usage patterns. It aims to reduce peak demand and overall energy consumption through measures like time-of-use pricing, demand response programs, and incentives for energy-efficient behavior.

Renewable Energy: Integrating renewable energy sources, such as solar, wind, hydro, or geothermal power, into energy systems is a key concept in energy management. It helps reduce reliance on fossil fuels, lowers greenhouse gas emissions, and promotes sustainable energy generation.

Energy Monitoring and Controls: Implementing energy monitoring systems and controls allows real-time tracking and analysis of energy consumption. This helps identify areas of high energy use, detect anomalies, and optimize energy usage through data-driven decision-making.

Energy Conservation: Energy conservation involves minimizing energy waste and unnecessary consumption by implementing various measures, such as insulation, efficient HVAC systems, energy-efficient building design, and behavioral changes like turning o↵ lights when not in use.

Energy Management Systems (EMS): EMS refers to the use of technology and software tools to monitor, control, and optimize energy consumption in buildings or industrial facilities. It includes features like energy data analysis, automated controls, and reporting functionalities to support e↵ective energy management.

Life Cycle Cost Analysis (LCCA): LCCA evaluates the total cost of a system or equipment over its entire life cycle, considering not just the initial investment but also operational costs, maintenance expenses, and energy consumption. It helps in making informed decisions by considering long-term financial and energy efficiency factors.

Energy Policy and Regulation: Energy management concepts also include understanding energy policies, regulations, and standards set by governments or industry bodies. These policies provide guidelines for energy efficiency, renewable energy integration, emissions reduction, and promote sustainable energy practices.

Related Work on Energy Management

The work in (13) discusses the integration of renewable energy resources, plug-in electric vehicles, and energy storage systems into the power grid to address the global energy crisis and promote sustainable practices. It emphasizes the need for an efficient energy management system (EMS) to regulate the flow of energy effectively. The paper highlights challenges posed by the variability and volatility of renewable energy sources, uncertainties associated with plug-in electric vehicles, fluctuating electricity prices, and changing load patterns. It stresses the role of the EMS in coordinating the sharing and trading of energy, ensuring cost-effective supply, and maintaining the reliability, security, and efficiency of the power system.
A comprehensive analysis of the EMS framework, objectives, architecture, benefits, and challenges is conducted. The paper explores distributed energy resources and programs like demand response and power quality management implemented within the EMS. It also examines uncertainty quantification methods and optimization techniques employed to achieve various objectives under multiple constraints. The paper provides recommendations for research and development to advance optimized energy management systems applicable to different domains. It serves as a valuable resource by offering a comprehensive analysis of stakeholders, distributed energy resources, programs, uncertainty quantification methods, and optimization techniques.
Additionally, this paper reviews the energy management systems in the context of the smart grid, discussing control architectures, stakeholders, uncertainties, and challenges. It emphasizes the importance of advanced metering, communication, and the Renewable Energy Management System (REMS). The need to address uncertainty, ensure security and privacy, and enhance cost-effective smart grid networks is highlighted. Overall, the paper underscores the significance of the EMS in achieving efficient and sustainable operation of the smart grid. It provides insights into various aspects of the EMS and offers recommendations for future research and development.

The research work in (14) introduces a novel software called cooperative energy management software (EMS) that aims to enhance the functioning of networked microgrids (MGs). The software explicitly models the cooperative behavior of MGs, allowing them to autonomously organize themselves into multiple stable coalitions. The primary objective of these coalitions is to facilitate efficient and cost-effective energy exchange among the participating MGs. Within each coalition, multiple MGs collaborate by engaging in energy trading with competitive pricing strategies, thus maximizing their individual utility.
To address the energy management problem in networked MGs, we propose a coalition

formation game between MGs. We develop a merge-and-split-based coalition formation (MSCF) algorithm, which ensures the stability of the formed coalitions and maximizes the profits of the MGs. Additionally, an intra-coalition energy transfer (ICET) algorithm is designed to enable energy exchange between MGs within the same coalition, minimizing power loss.

Through simulation experiments, we demonstrate the effectiveness of the proposed co-operative energy management software. The results reveal a significant improvement in profit maximization, exceeding 21%, and a substantial reduction in power loss, surpassing 51%. These findings highlight the considerable benefits of implementing the cooperative energy management software in networked microgrids.

The work in (15) discusses the components of a smart home system, including hardware, software algorithms, network connections, and sensors. It explains how these components work together to provide various services. The development of a smart grid enables residents to schedule their electricity usage and reduce costs. Energy management at the household level considers environmental impact and supports residents' lifestyles. This paper examines commercial and technical reasons for managing energy at the household level. Commercially, it allows passive residential customers to participate actively in the energy market. Technical aspects include peak shaving, valley filling, load shifting, flexible load curves, strategic conservation, and load growth. Socio-economic impacts are also important to consider.

It reviews the concept of energy management systems for residential customers, discussing technological approaches and major components. Concerns and challenges associated with smart technologies, including cost, implementation, and privacy issues, are discussed. The paper explores electricity pricing and demand-side management for residential customers. It highlights the inefficiency of current pricing models and introduces various pricing models to align retail prices with wholesale prices. The goal is to incentivize users to shift high-load appliances to off-peak hours, reducing costs and load demand. Demand-side management programs, including conservation, energy efficiency, and load management, are introduced. The need for practical solutions to shift high-power appliances to off-peak hours is emphasized. Demand response as a means to lower electricity use during high prices or reliability issues is discussed.

This work also presents a classification of demand response activities and favors incentive-based programs. An algorithm is proposed to manage power-intensive loads in a house based on peak reduction targets and user preferences. To implement demand response, a fully automated solution called auto-DR is necessary. Home Energy Management systems play a significant role in this regard. The paper concludes by emphasizing the importance of home energy management systems in reducing environmental impact and supporting human lifestyles. Technical and economic aspects are considered, and challenges are addressed. A framework for future energy management systems is proposed.

The work in (16) presents a multi-agent based solution for smart grid to improve the cyber-physical security. This work identifies false data injection attacks in smart grid by performing a real-time sensitivity analysis which check for abnormalities in the smart grid.

Conclusion

In conclusion, smart grids represent a significant advancement in the energy sector, offering numerous benefits for consumers, utilities, and society as a whole. As the demand for electricity continues to grow, the deployment of smart grids will play a vital role in ensuring a sustainable and reliable energy supply for the future.

In order to achieve better results in terms of energy cost and energy availability, AI-based solutions present a great potential for smart grid systems as a developped cyber-physical system. Artificial intelligence can play a pivotal role in the smart grid at their different levels to improve the overall performance of the grid.

2 Smart Grid and Artificial Intelligence

Introduction

The integration of Artificial Intelligence (AI) in the field of energy management has led to significant advancements in the development of smart grids. Smart grids, as the next generation of power systems, aim to optimize the generation, distribution, and consumption of electricity by leveraging intelligent technologies. With the increasing demand for clean and sustainable energy solutions, the application of AI in smart grids holds great promise for revolutionizing the energy sector. This essay explores the intersection of AI and smart grids, highlighting the benefits, challenges, and future prospects of this dynamic combination.

Artificial Intelligence

Artificial intelligence encompasses a broad domain that encompasses multiple perspectives and diverse definitions. In essence, it involves a range of techniques intended to create machines capable of performing specific tasks and addressing human challenges. Artificial intelligence can be seen as the integration of methodologies that enable the development of machines with problem-solving abilities, designed to fulfill specific objectives. According to Turing:"What makes it difficult to distinguish between a task performed by a human being or a machine".

AI has captured our imaginations and has been a topic of research since a group of computer scientists coined the term at the Dartmouth Conferences in 1956, giving birth to the field of AI. Throughout the years, AI has been praised as the key to a prosperous future for our civilization, while also being dismissed as a far-fetched idea by some skeptics in the tech industry. However, it wasn't until 2012 that AI started to gain traction and prove its potential. In recent years, AI has experienced a significant surge, particularly since 2015. This can be attributed to the widespread availability of GPUs, which have made parallel processing faster, more a↲ordable, and more powerful. Additionally, the explosion of data in various forms (thanks to the Big Data movement) such as images, text, transactions, and mapping data, combined with practically limitless storage capabilities, has contributed to this AI revolution.

Computer scientists transitioned from a period of relative disappointment before 2012, to a flourishing era where AI applications are now utilized by millions of people worldwide on a daily basis. During the summer conference of 1956, the vision of AI pioneers revolved around constructing sophisticated machines, utilizing the emerging computers of that time, which would possess human-like intelligence. This concept, often referred to as "General AI," involves creating extraordinary machines that not only replicate our senses but also possess reasoning abilities identical to our own. We have witnessed these machines portrayed extensively in movies, both as friendly companions like C-3PO and as formidable adversaries like The Terminator. However, General AI has remained confined to the realms of fiction and science fiction novels for a valid reason—we have not been able to achieve it yet, or at least not to the desired extent.

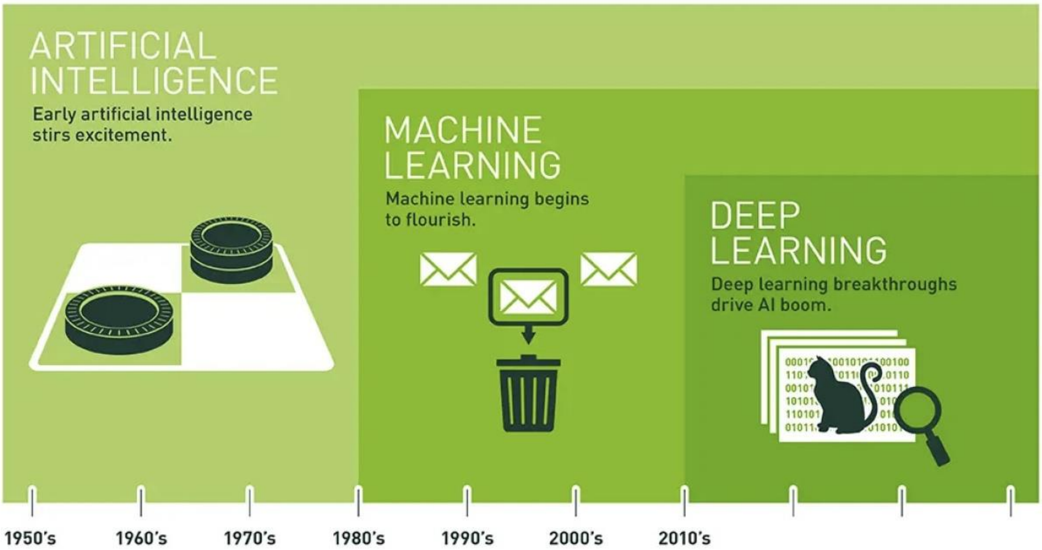


Figure 9: AI over time

Instead, what we have been able to accomplish thus far falls under the category of ”Narrow AI.” Narrow AI encompasses technologies capable of performing specific tasks as well as, if not better than, humans. Examples of narrow AI include image classification on platforms like Pinterest and facial recognition on Facebook.

These practical applications of Narrow AI demonstrate certain aspects of human intelligence. But how does it work? Where does this intelligence originate from? This brings us to the next concept, known as machine learning.

Machine Learning

At its core, machine learning involves the utilization of algorithms to analyze data, acquire knowledge from it, and subsequently make determinations or predictions about various aspects of the world. Instead of manually coding software routines with specific instructions to accomplish a particular task, machine learning takes a di↵erent approach. It involves training the machine by exposing it to extensive amounts of data and employing algorithms that enable it to learn how to perform the given task.

The concept of machine learning originated from the minds of early AI researchers. Throughout the years, various algorithmic approaches have been developed, including decision tree learning, inductive logic programming, clustering, reinforcement learning, and Bayesian networks, among others. However, none of these approaches managed to achieve the ultimate objective of General AI, and even achieving Narrow AI remained largely challenging with early machine learning methods.

Deep Learning

The advent of deep learning has revolutionized the practical applications of machine learning and has had a significant impact on the broader field of AI. Deep learning approaches have enabled the breakdown of complex tasks into manageable components, thereby making various forms of machine assistance not only feasible but also highly probable. Driverless cars, enhanced preventive healthcare measures, and even improved movie recommendations are already a reality or within close reach. AI is undeniably the present and the future, and with the assistance of deep learning, it may even bring us closer to the realm of science fiction that we have long envisioned. The idea of having a C-3PO-like companion is enticing, while the menacing presence of a Terminator can be set aside.

Artificial neural networks

Another algorithmic approach that emerged from the early machine learning era was artificial neural networks, which saw varying levels of popularity over the decades. Inspired by our understanding of the biological structure of our brains, artificial neural networks attempt to replicate the interconnections between neurons. However, unlike biological brains, where any neuron can connect with any other neuron within a certain physical proximity, artificial neural networks consist of discrete layers, connections, and directions of data flow.

For instance, in an image processing scenario, the image can be divided into multiple tiles that serve as inputs to the first layer of the neural network. Each individual neuron in the first layer processes the data and passes it on to the next layer. This sequential process continues through subsequent layers until the final layer produces the desired output.

Each neuron assigns a weight to its input, determining its relevance to the given task. The final output is then determined by the cumulative effect of these weightings. To illustrate with the example of a stop sign, attributes of an image, such as its octagonal shape, red color, letters, size, and motion, are examined by the neurons. The neural network's objective is to determine whether the image represents a stop sign or not. It generates a "probability vector," which is essentially an educated guess based on the assigned weightings. In our example, the system might indicate an 86% probability that the image is a stop sign, 7% probability for a speed limit sign, 5 % probability for a kite stuck in a tree, and so on. The network architecture then provides feedback to the neural network, indicating its accuracy.

However, it is worth noting that until recently, neural networks were largely disregarded by the AI research community. Despite existing since the early days of AI, they demonstrated limited success in achieving "intelligence." One of the main challenges was the significant computational intensity even for basic neural networks, making it impractical to employ them. Nonetheless, a small group of researchers led by Geoffrey Hinton at

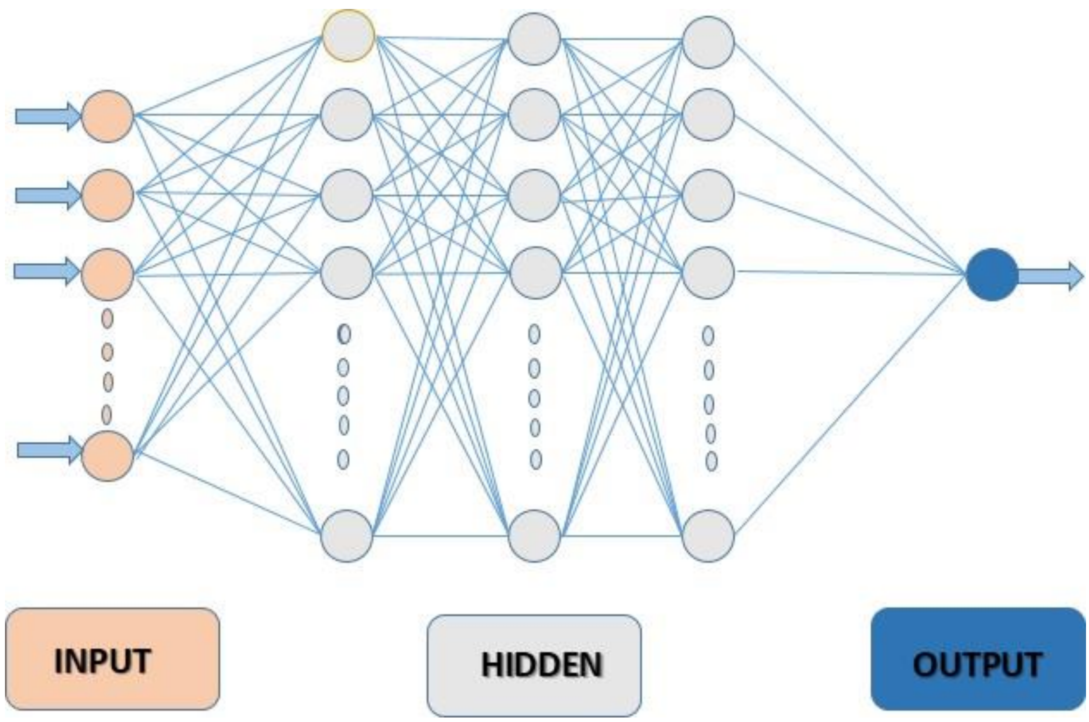


Figure 10: Overview of Artificial Neural Network

the University of Toronto persisted and eventually parallelized the algorithms to run on supercomputers, proving the concept. It was with the deployment of GPUs that the true potential of neural networks was realized.

Returning to our stop sign example, during the training phase, it is highly likely that the network will produce numerous incorrect answers. It requires training, exposure to hundreds of thousands or even millions of images, to precisely tune the weightings of neuron inputs, enabling it to provide accurate responses almost every time, regardless of fog, sunlight, or rain. At that point, the neural network has essentially taught itself to recognize a stop sign, or in the case of Facebook, your mother's face, or in Andrew Ng's case, a cat, as demonstrated in his work at Google in 2012.

Today, machine learning models trained through deep learning, particularly in image recognition tasks, can outperform humans in certain scenarios, ranging from identifying cats to detecting indicators of cancer in blood samples or tumors in MRI scans. For example, Google's AlphaGo learned the game of Go by playing against itself repeatedly, tuning its neural network through extensive self-training.

The simplest type of ANN is a perceptron. A perceptron has one input, one output, and one weight. The output of the perceptron is calculated as follows:

$$y = \sum w_i x_i + b \tag{1}$$

where:
y is the output of the perceptron
w is the weight
x is the input
b is the bias

The perceptron can be used to classify data. For example, we can use a perceptron to classify images of cats and dogs. We would train the perceptron on a set of images of cats and dogs. The perceptron would learn to associate the input features (e.g., the shape of the eyes, the length of the tail) with the output labels (cat or dog).

More complex ANNs can be created by connecting multiple perceptrons together. These networks are called multilayer perceptrons (MLPs). MLPs can be used to solve more complex problems, such as predicting the price of a stock or recognizing faces. The training of an ANN is an iterative process. The ANN is first initialized with random weights. Then, the ANN is presented with a set of training data. The ANN calculates its output for each training example. The error between the ANN's output and the ground truth label is then calculated. The weights of the ANN are then updated to reduce the error. This process is repeated until the error is minimized. ANNs have been shown to be very e↵ective at solving a wide variety of problems. They are used in a variety of applications, including: Image recognition, Natural language processing, Speech recognition, Medical diagnosis, Financial forecasting and Robotics.

ANNs are a powerful tool that can be used to solve a wide variety of problems. However, they can be difficult to train and require a large amount of data and need huge computational power.

Application of AI in Smart grid

Artificial intelligence (AI) holds immense potential in the field of smart grid, an advanced electrical power system that integrates renewable energy sources, energy storage, and intelligent control systems to enhance energy efficiency, reliability, and sustainability. Below are some key applications of AI in the smart grid:

Demand Response: AI enables the prediction of energy demand and optimization of power distribution to meet that demand. By employing machine learning algorithms, smart grid systems can analyze historical usage patterns and weather forecasts to forecast future energy demand. This predictive capability allows the grid to adjust power supply, minimizing energy loss and preventing blackouts.

Energy Forecasting: AI plays a crucial role in forecasting energy production from renewable sources like solar and wind energy. By analyzing weather patterns and other variables, machine learning algorithms can predict the amount of energy that will be generated by these sources.

Grid Management: AI facilitates the management and optimization of the power grid. Through AI algorithms, sensor data from smart meters, smart transformers, and other grid components can be analyzed to identify patterns and anomalies. This information aids in optimizing power distribution and mitigating the risk of blackouts.

Fault Detection and Diagnosis: AI assists in the detection of faults and diagnosis of problems within the smart grid. Machine learning algorithms analyze sensor data to identify grid faults such as power outages or voltage spikes. This data enables the identification of the root cause of the problem and facilitates necessary corrective actions.

Energy Trading: AI optimizes energy trading within the smart grid. By analyzing market data, machine learning algorithms identify profitable trades and provide recommendations to energy traders, enhancing the efficiency of energy trading operations.

In summary, AI demonstrates significant potential in leveraging smart grid technology, enhancing its capabilities in demand response, energy forecasting, grid management, fault detection and diagnosis, as well as energy trading.

Related Work

The work in (17) explores the role of artificial intelligence (AI) in achieving energy sustainability in smart cities. It focuses on the use of smart metering and non-intrusive load monitoring (NILM) to optimize energy consumption. The paper proposes a genetic algorithm support vector machine multiple kernel learning (GA-SVM-MKL) approach for NILM, which achieves high accuracy in appliance classification.

The research integrates insights from AI, IoT, and big data analytics, emphasizing their potential in addressing energy sustainability challenges. It highlights the significance of microgrid applications and smart metering in smart grid development. The paper calls for increased attention from stakeholders and suggests future research directions, including the use of blockchain technology and the analysis of user behavior in smart energy modeling.
Overall, it contributes to the understanding of energy sustainability in cities and provides insights into utilizing AI for energy optimization in smart cities.

The research work in (18) discusses the integration of artificial intelligence (AI) in smart cities to address various challenges. It analyzes 133 research articles published between 2014 and 2021, focusing on domains such as healthcare, education, environment, mobility, agriculture, risk management, and security.
The findings highlight the significant impact of AI adoption in smart cities, particularly in healthcare, mobility, privacy and security, and energy sectors. The healthcare industry has seen a substantial increase in AI-based advancements since the COVID-19 pandemic. The study identifies AI algorithms such as ANN, RNN/LSTM, CNN/R-CNN, DNN, and SVM/LS-SVM as influential in smart city domains. AI offers benefits such as automation, reduced errors, data-driven decision-making, and improved urban management. However, it also presents regulatory challenges and risks, including privacy concerns and ethical considerations.
This research emphasizes the importance of carefully addressing these challenges and risks while integrating AI technologies in smart cities. It concludes that AI can greatly benefit smart cities, but proper attention must be given to regulatory issues and overcoming barriers to implementation.

The study in (19) focuses on the importance of developing Intelligent Transport Systems (ITS) in smart cities in India to address various urban challenges. Issues such as inefficient public transport, congestion, road accidents, inadequate parking, and rising energy costs are prevalent in Indian cities. To tackle these problems, the study suggests the adoption of Artificial Intelligence (AI) in developing smart public transport, intelligent traffic management, smart traveler information systems, smart parking management, and safe mobility and emergency systems.
The authors highlight the urgency of implementing AI-based ITS in smart cities to overcome these challenges. It emphasizes the role of AI in optimizing resource utilization, controlling operations, and improving efficiency in transportation systems. The adoption of AI in traffic management can significantly reduce fuel consumption, travel time, stops, and carbon emissions.
This study provides valuable insights for engineers and planners involved in developing efficient transportation systems in smart cities in India. It underscores the need for AI applications in various sub-systems of ITS and emphasizes the collaborative efforts of transportation experts and computer engineers. Overall, the study concludes that the adoption of AI-based ITS is crucial for the development of smart cities in India to create accessible, safe, environmentally friendly, and efficient transport systems.

The research in (20) focuses on the challenges and implications of incorporating artificial intelligence (AI) in the smart energy domain, specifically in the context of renewable energy (RE) in the European Union (EU). The study emphasizes the need to optimize energy systems, develop customizable networks, and leverage AI and machine learning techniques to address these challenges.
The authors highlight the significance of RE in global development, considering climate change and resource depletion. AI presents new opportunities for organizing activities and meeting the evolving demands of the energy sector. The aim is to improve the design, deployment, and production of RE to overcome obstacles that affect the sector's growth and resilience.
Specific analyzed areas include the efficiency of transforming RE in the energy chain, the structure of renewable energy sources, labor productivity in the RE sector, and its relationship with investments. The research also explores the implications of AI adoption for RE within the context of future smart cities research.
The main contributions of this study are the development of a framework to understand the role of AI in the RE sector in Europe and a discussion of the implications for future smart cities research. The research focuses on the integration of AI technology in smart energy systems and grids, emphasizing the need for a comprehensive understanding of computational, economic, and social aspects. It examines the implications of AI and disruptive technologies for economic models, with a particular focus on energy management in the RE sector and the adoption of disruptive technologies.
The research also identifies potential directions for future research in this field.
Overall, the research provides insights into the potential of AI in enhancing renewable energy efficiency, sustainability, and the development of smart energy infrastructure. The paper concludes by summarizing key findings and suggesting directions for future research in the field of smart cities. Despite advancements, barriers persist in the widespread implementation of RE, extending beyond technology to include policy-making. This highlights the need for further studies and debates to address these barriers effectively.

Conclusion

In conclusion, the integration of Artificial Intelligence (AI) in smart grids has emerged as a transformative force in the energy sector. By harnessing the power of AI algorithms and data analytics, smart grids have the potential to optimize energy generation, distribution, and consumption, resulting in improved efficiency, reliability, and sustainability.

As technology continues to advance, and stakeholders collaborate to address the associated challenges, we can expect to witness a future where AI-powered smart grids play a vital role in delivering efficient, reliable, and sustainable energy systems. By embracing the potential of AI and smart grids, we can pave the way for a greener and smarter energy future.

3 System Model

Introduction

The electric power systems usually consist of three tiers, namely the generation level (comprising large power stations like nuclear or thermal power plants), the distribution level (including distribution substations), and the consumption level (consisting of smart homes and buildings). Each of these levels in the smart grid is characterized by specific voltage levels: High (75kV) for generation, Medium (30kV) for distribution, and Low (400V) for consumption. These levels encompass various electrical devices such as generators, transformers, and electrical lines that interconnect the di↵erent components of the smart grid. Figure 1 provides a visual representation of the smart grid levels.
In this project, we will focus on the distribution and consumption level where we implement two di↵erent solutions for energy management.

Electric power distribution represents the ultimate phase of delivering electricity, transporting it from generation stations or transmission substations to individual end-users. These end-users receive their power through distribution substations that link the transmission system to the distribution system with the aid of transformers. Medium voltage power is transmitted from the primary distribution lines to distribution transformers situated in close proximity to consumers. These distribution transformers reduce the medium voltage to a suitable level for household appliances and lighting purposes. Consumers receive their power supply from these transformers via secondary distribution lines. High-demand consumers requiring significant amounts of electricity are directly connected to the primary distribution level or sub-transmission level.

The transition from the transmission system to the distribution system occurs within power substations, which serve several important functions. These functions include the use of switches and circuit breakers to disconnect the substation from the transmission system or to disconnect distribution lines. Additionally, transformers are employed to reduce the voltage, typically from 35 KV or higher, to distribution voltages. Primary distribution voltages, ranging from 4 to 35 kV, are primarily utilized to supply power to large consumers. However, the majority of utility consumers are connected to a transformer that further lowers the voltage to a lower utilization voltage.

Figure.11 shows the architecture of the proposed AI-based solution for smart grid energy management.

Smart grid mathematical formulation : Distribution lvevel

In this section, we will formulate the smart grid system at the distribution level.
In the nonlinear smart grid formulation, the power balance equations at bus i and time t are given by (21)

$$P^{t}_{Gi} + P^{t}_{Si} - P^{t}_{Di} = 0 \tag{2}$$

$$Q^{t}_{Gi} + Q^{t}_{Si} - Q^{t}_{Di} = 0 \tag{3}$$

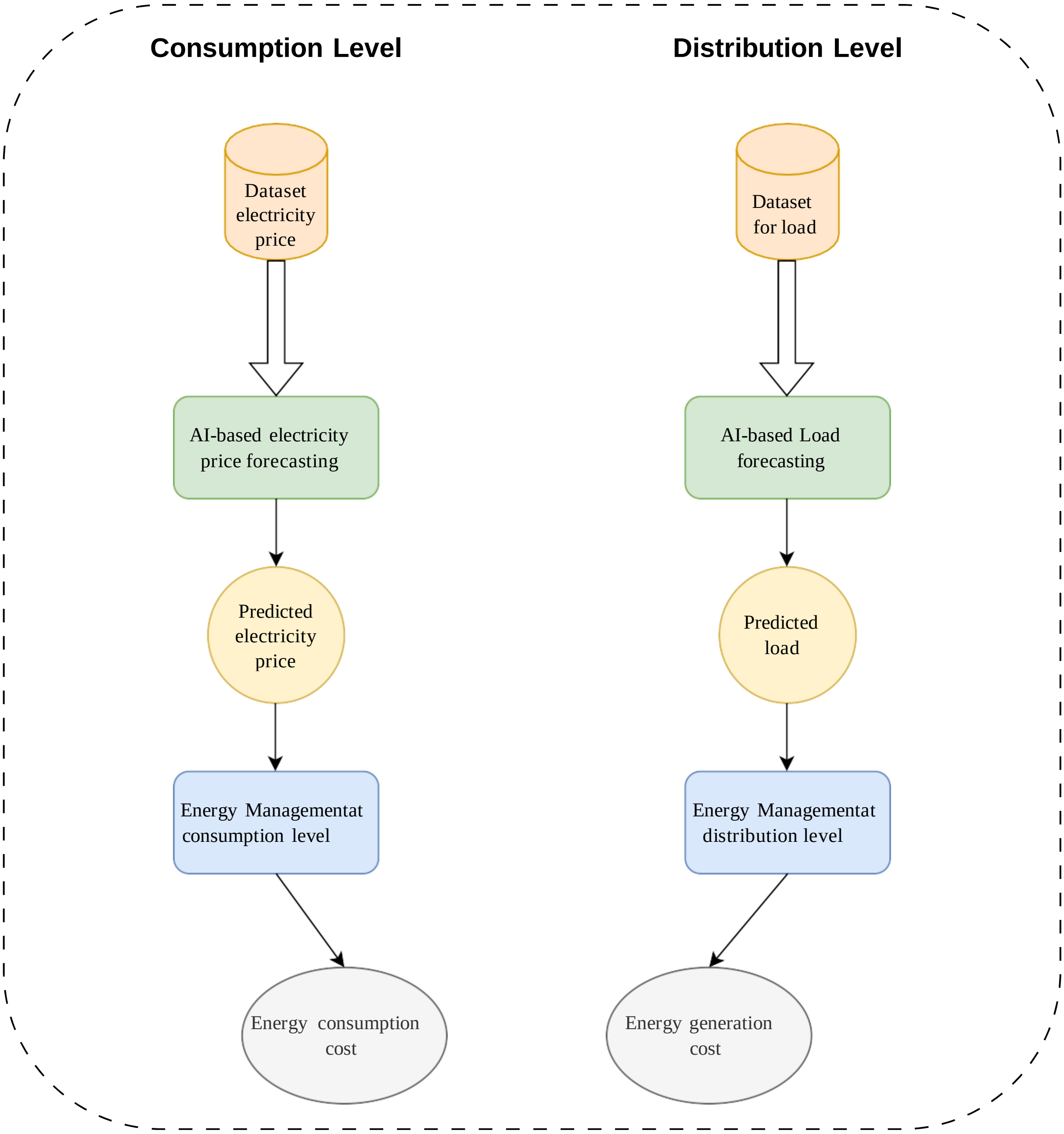


Figure 11: Architecture of the proposed solution

where the P^t_{Gi}, P^t_{Si} and P^t_{Di} are the active power generation, active power from energy storage systems and active power demand, respectively. Q^t_{Gi}, Q^t_{Si} and Q^t_{Di} are the reactive power generation, reactive power from energy storage systems and reactive power demand, respectively. The power loss in transmission lines is calculated as follows (21):

$$P_L(i,e) = I^2 R = \left[\frac{P(E)}{V}\right]^2 \cdot \rho \cdot d(i,e) \tag{4}$$

Objective function

The standard optimal power flow minimizes the objective function f which is simply a summation of individual polynomial cost functions of real and reactive power injections, respectively, for each generator (21):

$$\min f(x) = \min_{\Theta, V_m, P_g, Q_g} \sum_{i=1}^{n_g} f_P^i(p_g^i) + f_Q^i(q_g^i) \tag{5}$$

Optimization variables

For the standard optimal power flow problem, the optimization vector x consists of the n_b x 1 vectors of voltage angles Θ and magnitudes V_m and the n_g x 1 vectors of active and reactive power injections P_g and Q_g of generators (21):

$$\mathbf{x} = \begin{bmatrix} \Theta \\ V_m \\ P_g \\ Q_g \end{bmatrix} \tag{6}$$

Constraints

The complex voltages on buses are characterised by a variable voltage magnitude V_i^t which is maintained within its limits as follows (21):

$$\underline{V_i} \leq V_i^t \leq \overline{V_i} \tag{7}$$

The limitation on line current in transmission line l between buses i and j is also imposed as follows (21):

$$I_{ij}^t \leq \overline{I_{ij}} \tag{8}$$

Proposed solution in Distribution level : Load forecasting approach

Load forecasting based on deep learning approach

Load forecasting plays a vital role in the electrical power systems domain as it enables utility companies and grid operators to effectively plan and optimize their operations. Deep learning methods have demonstrated significant potential in load forecasting due to their ability to capture intricate patterns and dependencies present in historical load data. Here’s a summarized overview of load forecasting using deep learning:

Data Preparation: The initial step involves gathering historical load data, typically recorded at regular intervals (e.g., hourly or daily) over a considerable time span. Additional relevant data, such as weather conditions or holidays, may also be considered. The data is then preprocessed to handle missing values, outliers, and normalization, ensuring consistency and enhancing model performance.

Model Architecture: Deep learning models like recurrent neural networks (RNNs) or long short-term memory (LSTM) networks are commonly utilized for load forecasting. These models are designed to capture temporal dependencies and patterns in sequential data. Convolutional neural networks (CNNs) can also be employed to extract spatial and temporal features from load data.

Training: The prepared data is divided into training and validation sets. The model is then trained using the training set, where it learns the patterns and relationships present in the historical load data. During training, the model optimizes its internal parameters by minimizing a loss function, often a mean squared error (MSE) or mean absolute error (MAE) between the predicted and actual load values.

Model Evaluation: Following training, the model is evaluated using the validation set to assess its performance. Various metrics, such as root mean squared error (RMSE), mean absolute percentage error (MAPE), or correlation coefficient, are employed to measure the accuracy and reliability of the load forecasts.

Testing and Deployment: Once the model's performance is deemed satisfactory based on the evaluation results, it can be deployed to make load forecasts on new, unseen data. The model takes historical load data as input and generates predictions for future time intervals. These forecasts serve as valuable guidance for decision-making processes, including load scheduling, resource allocation, and energy market operations.

Data Analysis

Electricity usage plays a crucial role in both production and everyday life, making any factors related to human activities in these areas influential in determining the demand for power. In this section, we examine significant factors that have a substantial impact on power consumption. By analyzing these factors, we gain valuable insights into how they a↵ect load forecasting, allowing us to incorporate them e↵ectively into our Deep Neural Network model.
The short-term load data of a specific region exhibits noticeable periodicity, primarily driven by the regular work and lifestyle patterns of individuals. Figure.12 illustrates the periodic variation of hourly load values from the dataset spanning 2014 to 2022. Furthermore, Figure 12 demonstrates that the load data curves for three consecutive weeks exhibit similar patterns that align with one another. As a result, historical load usage data serve as reliable indicators for load prediction.
In addition, it can be seen from Figure.13 that load data curves of three consecutive weeks shows similar patterns aligning with each other accordingly. Thus, historical load usage data serve as robust indicators for load prediction.

The impact of weather on electricity consumption is substantial, particularly in relation to people's work schedules. Temperature, in particular, plays a significant role, as unfavorable temperatures often lead to higher energy usage for heating or cooling, as well

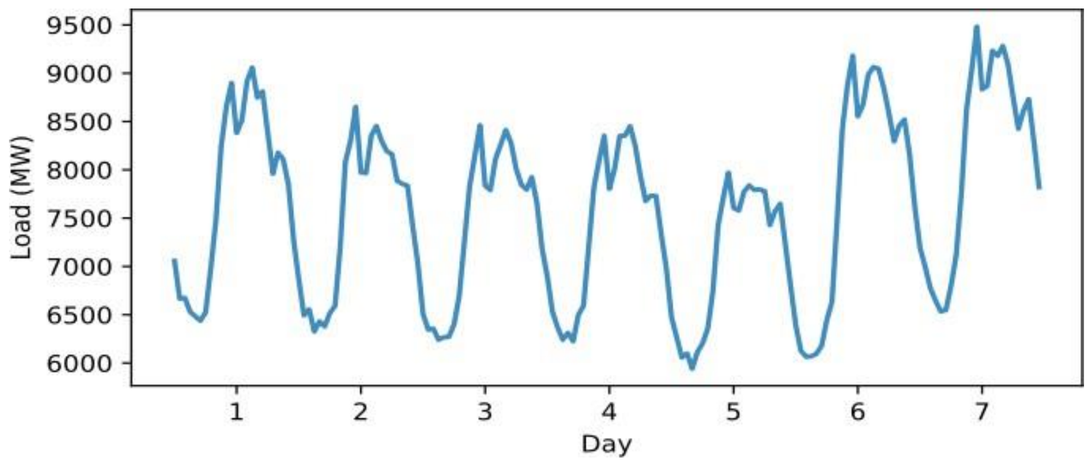


Figure 12: Daily periodicity (22)

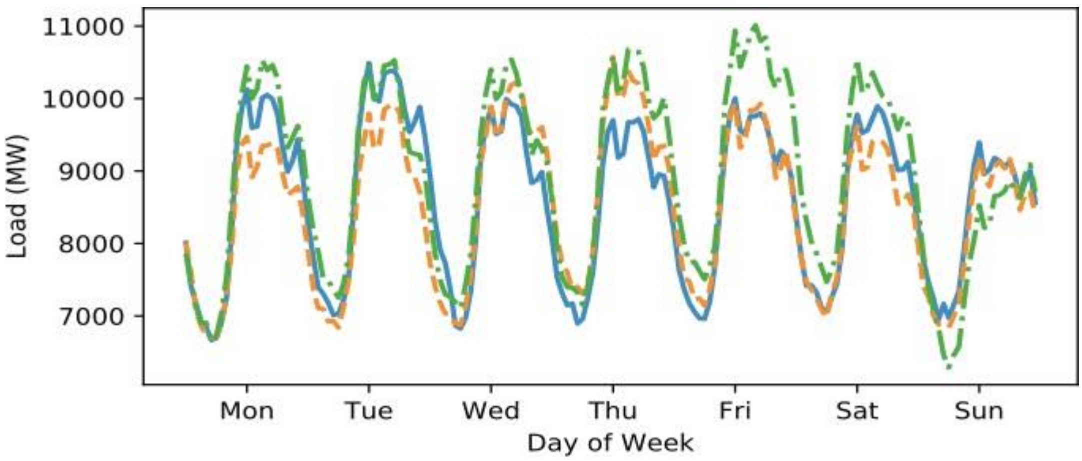


Figure 13: Weekly periodicity (22)

as increased indoor activities. Figure.14 demonstrates a clear correlation between the minimum daily temperature and the peak load. Additionally, other weather factors such as humidity and wind speed could also provide valuable insights for load forecasting.

Additionally, when predicting hourly load, the hour of the day itself and the corresponding day of the week are evident indicators of power consumption. Therefore, we incorporate these types of features into our model. The maximum and minimum temperatures, the hour of the day, and the day of the week features are utilized as inputs for the component of our model.

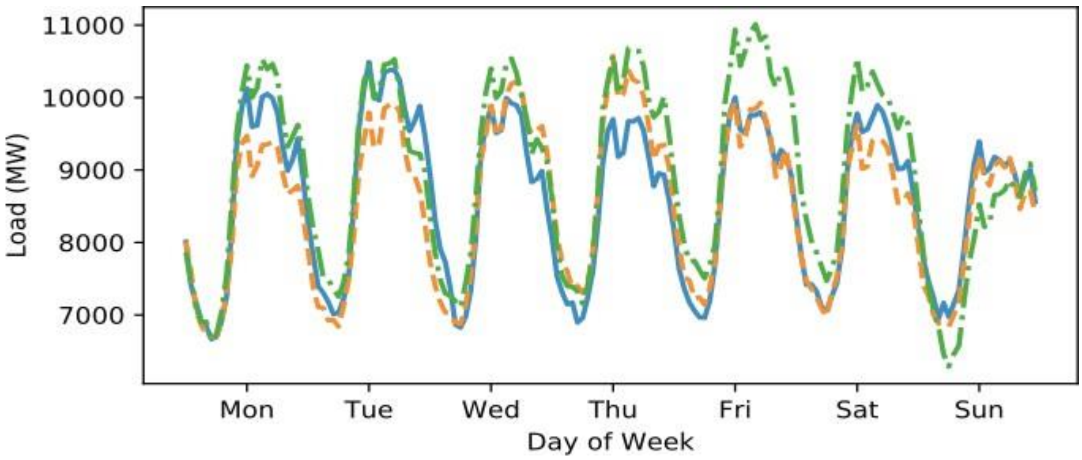


Figure 14: Daily minimum temperature (22)

Deep Neural Network for Load Forecasting

Deep neural networks (DNNs) are a type of machine learning algorithm that can be used for load forecasting. DNNs are able to learn complex patterns in data, and they have been shown to be very e↵ective for load forecasting tasks.

A DNN is a type of artificial neural network (ANN) that has multiple layers between the input and output layers. This allows the DNN to learn more complex patterns in the data than a shallow ANN.

DNNs work by passing the input data through a series of layers. Each layer contains a number of neurons, and each neuron performs a simple calculation on the data. The output of each neuron is then passed to the next layer, and so on.

DNNs can learn complex patterns in data. This allows them to make more accurate predictions than traditional forecasting methods. DNNs can be trained on large datasets. This allows them to learn the patterns in load demand even if the data is noisy or incomplete. DNNs can be used to model both short-term and long-term load dynamics. This makes them a versatile tool for load forecasting applications.

di↵erent DNN architectures that can be used for load forecasting, and some of the common architectures include Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Hybrid DNNs.

Convolutional Neural Networks (CNNs): CNNs are commonly used for image processing tasks but can also be applied to sequential data like load forecasting. They excel at learning hierarchical representations and capturing spatial or temporal patterns. In load forecasting, CNNs can learn to extract features from historical load data, taking into account the temporal relationships and patterns.

Recurrent Neural Networks (RNNs): RNNs are well-suited for sequential data analysis and can e↵ectively model long-term dependencies in time series data. RNNs have recurrent connections that allow information to be persistently passed from one step to the next. This makes them suitable for capturing the temporal dynamics and patterns in load demand data.

Hybrid DNNs: Hybrid DNN architectures combine the strengths of both CNNs and RNNs. By using a combination of convolutional and recurrent layers, these models can capture both local and global features from load data and model short-term as well as long-term load dynamics. Hybrid DNNs have been shown to be e↵ective for load forecasting tasks, leveraging the advantages of both CNNs and RNNs.

Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are neural networks consisting of neuron-like nodes arranged in consecutive layers, resembling the structure of standard Neural Networks. Similarly to standard Neural Networks, RNNs have input layers, hidden layers, and output layers, with each connection between neurons having an associated trainable weight.
However, the key distinction lies in the assignment of neurons to specific time steps within RNNs. Each neuron is allocated to a fixed time step, and the neurons in the hidden layer are directed in a time-dependent manner. This means that each neuron in the hidden layer is fully connected only to the neurons in the same assigned time step, and it is connected unidirectionally to every neuron assigned to the subsequent time step. The input and output neurons solely connect to the hidden layers associated with their respective assigned time step.

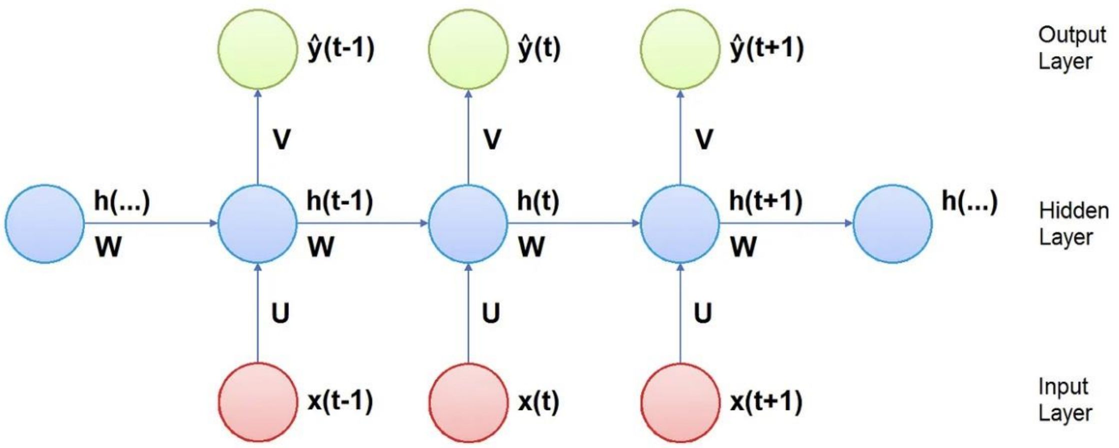


Figure 15: Architecture of RNN model

Due to the hidden layer's output from one time step being utilized as the input for the subsequent time step, neuron activation within RNNs is computed sequentially according to time order. At any given time step, only the neurons assigned to that particular time step calculate their activation.

Weights : In RNNs, the input vector at time t is linked to the hidden layer neurons of time t using a weight matrix U. The hidden layer neurons are connected to the neurons of time t-1 and t+1 through a weight matrix W. Additionally, the hidden layer neurons are connected to the output vector of time t via a weight matrix V. Importantly, all these weight matrices remain constant across each time step.
Input : The vector x(t) represents the network's input at time step t.
Hidden state : The vector h(t) signifies the hidden state at time t, serving as a form of network memory. It is computed based on the current input and the hidden state from the previous time step:
Output — The vector y(t) is the output of the network at time t:

Long Short Term Memory (LSTM) Neural Networks

LSTM, which stands for Long Short-Term Memory, is a type of Recurrent Neural Network (RNN) that is specifically designed to handle sequential data. Its unique architecture addresses the issue of vanishing gradients in traditional RNNs by introducing memory cells and gates to control the flow of information.
LSTM is widely employed in deep learning due to its capability to capture long-term dependencies in sequential data. This makes it particularly suitable for tasks like speech recognition, language translation, and time series forecasting, where the context of earlier data points can influence later ones.
Let's consider an example of using an LSTM network to predict car sales. Suppose we have access to historical data on monthly car sales over several years. Our goal is to utilize this data to make predictions about future car sales. To achieve this, we would train an LSTM network on the historical sales data, enabling it to forecast the next month's sales based on the previous months.
At each time step, the LSTM network takes the current monthly sales and the hidden state from the previous time step as input. It processes this information through its gates, updates its memory cells, and generates an output that predicts the next month's sales. To make the task more challenging, we can incorporate additional variables, such as average temperature and fuel prices, into the network's input. These variables can also impact car sales, and including them in the LSTM algorithm can enhance the accuracy of our predictions.
LSTM possess a remarkable capacity to represent sequential information and grasp extensive relationships, rendering them exceptionally well-suited for time series forecasting tasks such as anticipating sales figures, stock prices, and energy consumption.

Understanding LSTM Cells

The LSTM model architecture is as follows: Xt represents the input at a specific time step, ht is the output, Ct denotes the cell state, ft represents the forget gate, it corresponds to the input gate, Ot represents the output gate, and $\ddot{\text{A}}^t$ refers to the internal cell

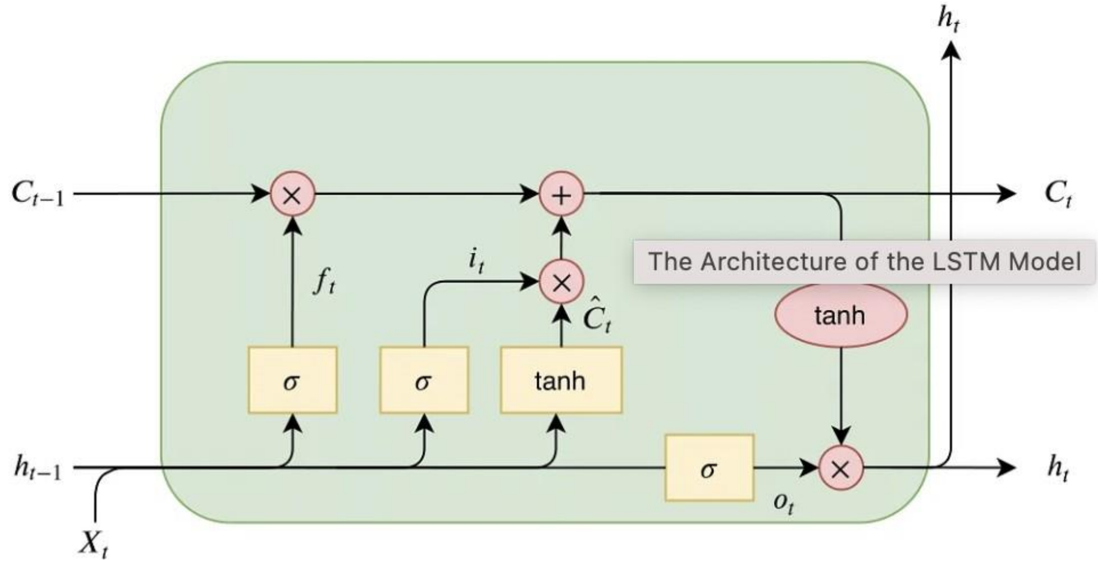


Figure 16: Architecture of LSTM model

state.
Long Short-Term Memory (LSTM) neural networks employ a set of gates to regulate the flow of information in a sequence of data. The forget, input, and output gates act as filters and operate as distinct neural networks within the LSTM architecture. They control how information enters the network, is stored, and eventually released.

1. Cell state: It preserves information across the entire sequence and serves as the network's memory.
2. Forget gate: This gate determines which information from previous time steps is relevant to retain.
3. Input gate: The input gate decides which information from the current time step should be incorporated.
4. Output gate: This gate determines the value of the output at the current time step.

The Forget Gate

The initial phase in the architectural design involves the Forget Gate, where the LSTM neural network assesses the relevance of elements in the cell state (long-term memory) based on the previous hidden state and new input data.
By feeding the previous hidden state (ht-1) and the new input data (Xt) into a neural network, a vector is produced. Each element of the vector represents a value ranging from 0 to 1, achieved through the utilization of a sigmoid activation function. The neural network within the forget gate is trained to generate values close to 0 for irrelevant information and values close to 1 for relevant information. These vector elements can be viewed as filters that permit more information when the value approaches 1.

$$\mathbf{f}_t = \sigma(W_f.[h_{t-1}] + b_f) \tag{9}$$

Subsequently, these output values are multiplied element-wise with the previous cell state (Ct-1). As a result, irrelevant portions of the cell state are down-weighted significantly,

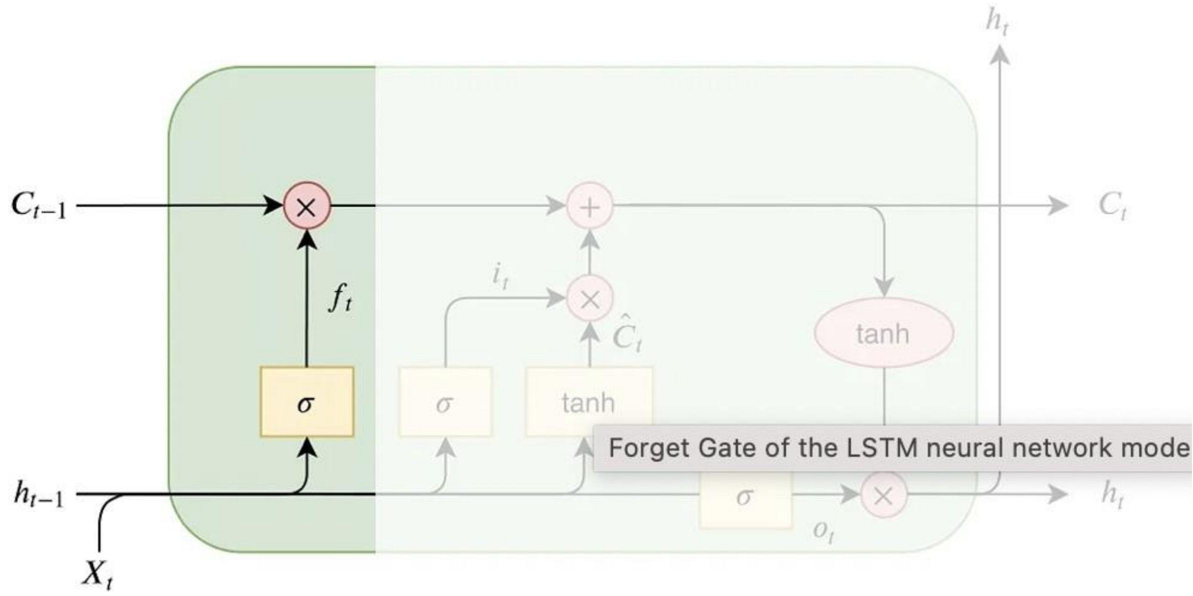


Figure 17: The Forget Gate

reducing their impact on subsequent steps.
In summary, the forget gate determines which components of the long-term memory should be discarded, based on the preceding hidden state and the new input data in the sequence.

The Input Gate

In the following phase, the input gate and the new memory network come into play. The primary objective of this step is to determine which fresh information should be integrated into the long-term memory (cell state) of the network, taking into account the previous hidden state and the current input data. Both the input gate and the new memory network function as independent neural networks, receiving identical inputs—the previous hidden state and the current input data. It is noteworthy that these inputs are the same inputs provided to the forget gate.

$$i_t = \sigma(W_i.[h_{t-1}, X_t] + b_i) \tag{10}$$

The input gate is a neural network that utilizes the sigmoid activation function. It serves as a filter to discern the valuable components within the new memory vector. By applying the sigmoid activation, it generates a vector of values within the range of [0,1]. This characteristic enables it to function as a filter through pointwise multiplication. Similar to the forget gate, a low output value from the input gate indicates that the corresponding element of the cell state should remain unaltered and not be updated.

New Memory Network

The neural network responsible for new memory generation utilizes the tanh activation function and has been trained to produce a ”new memory update vector” by combining the previous hidden state and the current input data. This vector encapsulates information from the input data while considering the contextual information provided by the previous hidden state. It indicates the necessary adjustments to be made to each component of

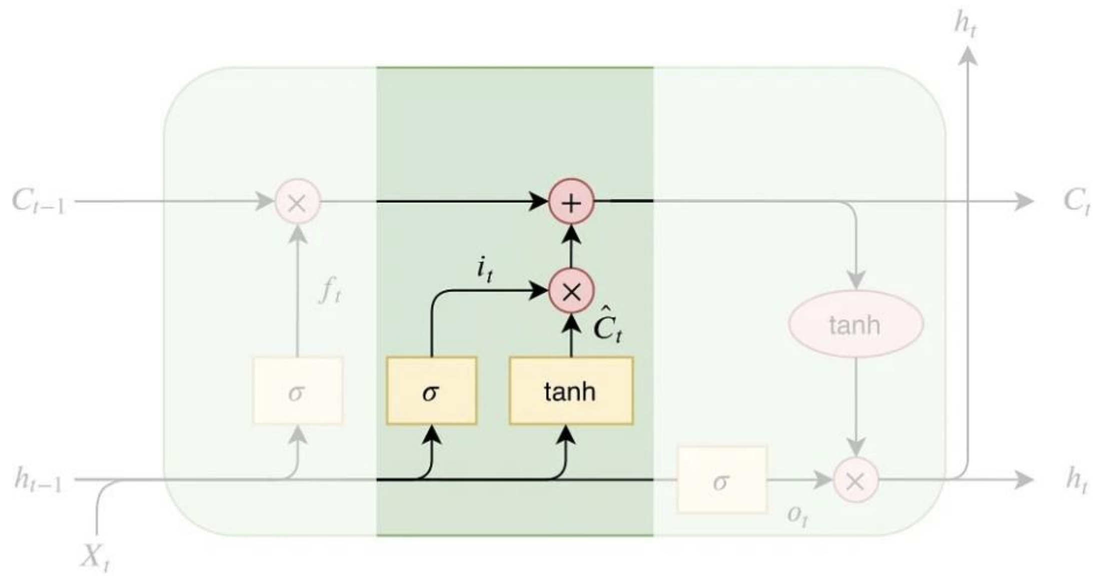


Figure 18: The Input Gate

the long-term memory (cell state) based on the most recent data.

$$\dot{C}_t = \tanh(W_C.[h_{t-1}, X_t] + b_C) \tag{11}$$

The tanh activation function is employed due to its output values ranging from -1 to 1. This range, including negative values, is crucial in reducing the impact of a component in the cell state.
However, it's important to note that the new memory vector alone does not determine the significance of retaining the new input data, necessitating the presence of an input gate.
To update the cell state, representing the LSTM network's long-term memory, the combination of the new memory update and the input gate filter is utilized. Pointwise multiplication governs this process, ensuring that only the relevant components of the new memory update contribute to the cell state.
Consequently, the updated cell state reflects the modified long-term memory of the network. The internal state is updated accordingly, adhering to this rule:

$$C_t = i_t.\dot{C}_t + f_t.C_{t-1} \tag{12}$$

The Output Gate

During the ultimate phase of an LSTM process, the determination of the new hidden state is accomplished by considering the updated cell state, previous hidden state, and new input data. This decision-making task is executed by the output gate.
The output gate plays a crucial role in defining the final hidden state of the LSTM network. It takes the updated cell state, previous hidden state, and new input data as inputs.

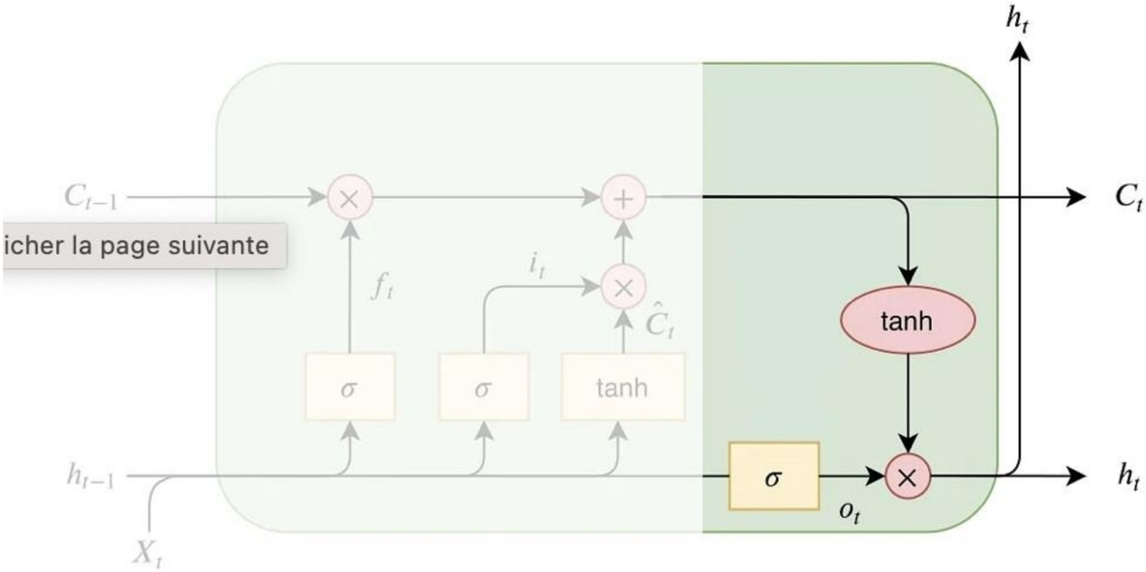


Figure 19: The Output Gate

Directly outputting the updated cell state alone would lead to excessive information disclosure. Hence, an output gate is employed as a filter.
The output gate functions as a neural network activated by the sigmoid function. It acts as a filter, discerning the relevant elements within the updated cell state that should be manifested as the new hidden state. The input for the output gate aligns with the previous hidden state and new data, and the sigmoid activation is employed to produce outputs ranging from 0 to 1.

$$o_t = \sigma(W_o.[h_{t-1}, X_t] + b_o) \tag{13}$$

The updated cell state subsequently undergoes a tanh activation to restrict its values within the range of -1 to 1. This squished cell state is then subjected to pointwise multiplication with the output of the output gate network. The outcome of this operation yields the final new hidden state.

$$h_t = o_t.\tanh(C_t) \tag{14}$$

Similar to other neural networks, the LSTM cell employs weight matrices and biases in conjunction with gradient-based optimization to learn its parameters. These parameters are associated with each gate, as evident in the provided equations. Specifically, the weight matrices are denoted as Wf, bf, Wi, bi, Wo, bo, and WC, bC, respectively.
In conclusion, the conclusive step of determining the new hidden state involves subjecting the updated cell state to a tanh activation to compress it within the range of -1 to 1. Subsequently, the previous hidden state and current input data are fed into a sigmoid-activated network to generate a filter vector. This filter vector is then multiplied pointwise with the compressed cell state to obtain the new hidden state, which serves as the output of this stage.

Unrolling LSTM Neural Network Model over Time

In the given architecture, the output gate represents the final stage within an LSTM cell, constituting just a single component of the overall process. Prior to obtaining the desired predictions from the LSTM network, several additional factors must be taken into account.

For instance, when attempting to forecast the stock price for the following day utilizing the past 30 days of pricing data, the steps within the LSTM cell would be reiterated 30 times. Consequently, the LSTM model would have sequentially generated 30 hidden states to forecast the stock price for the subsequent day. LSTM operates by propagating

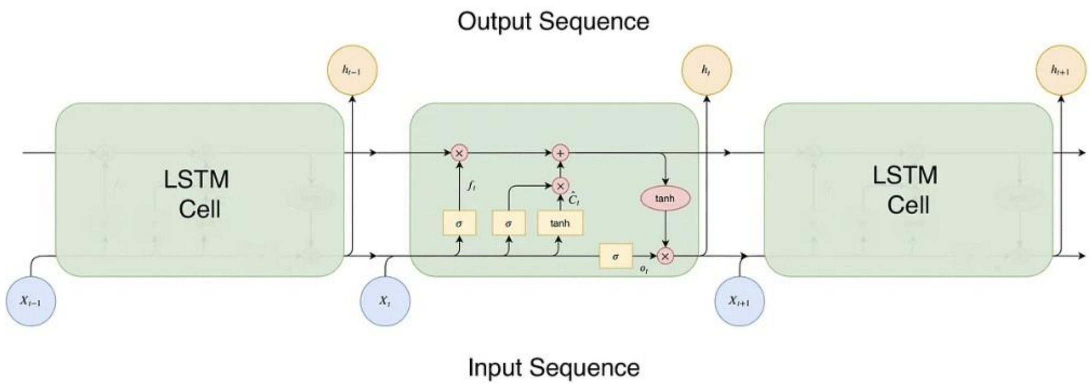


Figure 20: LSTM hidden states

information recurrently, forming a chain-like structure. The control over the transfer of the most recent cell output to the final state is achieved through the output gate. However, it is important to note that the output of the LSTM cell represents a hidden state and does not directly correspond to the stock price we aim to predict. To transform this hidden state into the desired output, a linear layer is applied as the concluding stage in the LSTM process. This linear layer step occurs only once, at the very end, and is not depicted in LSTM cell diagrams since it occurs after the repeated steps of the LSTM cell. The concept of unrolling LSTM models across time refers to the expansion of an LSTM network over a sequence of time steps. During this process, the LSTM network is essentially duplicated for each time step, and the outputs from one time step are used as inputs for the subsequent time step.

Smart grid mathematical formulation : Consumption level

The smart grid presents a complex network as a result of the intermittent nature of energy sources and energy demands. This complexity poses a challenge in solving the energy management problem, which is known to be NP-hard. To address these issues in the smart grid, it is crucial to consider the implementation of a nonlinear model.

System characteristics

In this section, we will formulate the smart grid system at the consumption level.
We model the smart grid as a connected graph $G = (N, L)$ where it is composed of a set of network buses denoted by $N = 1, 2, ..., n$. These buses are connected by a set of transmission lines denoted by $L = 1, 2, ..., n_l$.
In the proposed Energy management system, each consumer is characterized by its planned daily tasks TS_j. Let $TS = \{TS_1, TS_2, ..., TS_k\}$ be the set of daily tasks to be executed, each task can be characterized by two vectors (23):

$$X_j = \left[x_j^1, x_j^2, ..., x_j^t\right] \tag{15}$$

$$Y_j = \left[P_j \quad D_j \quad ST_j \quad FT_j \quad STP_j \quad FTP_j\right] \tag{16}$$

where

- X_j is the power consumption profile of the electric appliance that executes the task j where x_j^t is the power consumption of task j at time t.
- Y_j is the vector which characterizes the task j.
- $P_j = \sum_{t=1}^{T} x_j^t$ is the energy demand of the task j where the time horizon $T = 24$.
- D_j is the duration of the task j.
- $[ST_j, FT_j]$ are the earliest start time and finishing time to run the task **j** that define its admitted interval of execution.
- $[STP_j, FTP_j]$ is the time preferred window of the consumer to run the task **j**.

Let P_{ij}^t denotes the energy consumption of a consumer i for task j at time t. The total energy consumption of a consumer i at time slot t is defined as follows:

$$l^t = \sum_{j=1}^{k} P_{ij}^t \tag{17}$$

Consider $C_u^t(l^t)$ the quadratic cost function of utility grid at time slot t. The cost function of a consumer is defined as follows:

$$\sum_{t=1} C_u^t(l^t) = a^t(l^t)^2 + b^t l^t + c^t \tag{18}$$

Constraints

Time constraints

Each task j of duration D_j must be executed exactly once between its earliest start time ST_j and finishing time FT_j. The start time t_j of task j satisfies the following constraint:

$$ST_j \leq t_j \leq FT_j - D_j \tag{19}$$

Energy balance constraints

The energy generated at time t must be equal to the total energy consumed by satisfying:

$$E_g^t - E_c^t = 0 \tag{20}$$

where E_g^t is the energy produced by the utility grid and E_c^t is the total energy consumed.

Objective function

The main objective at consumption level is to reduce electricity bill. The objective function is given as follows:

$$\min_{C} J = \sum_{t=1}^{T} {}_c^t \tag{21}$$

Proposed solution in Consumption level : Electriciy price forecasting

Electricity Price forecasting based on Deep learning approach

Within this section, we provide a detailed explanation of our advanced approach, which is based on time series forecasting solution using LSTM model. The fundamental architecture of our model is illustrated in Figure 21.

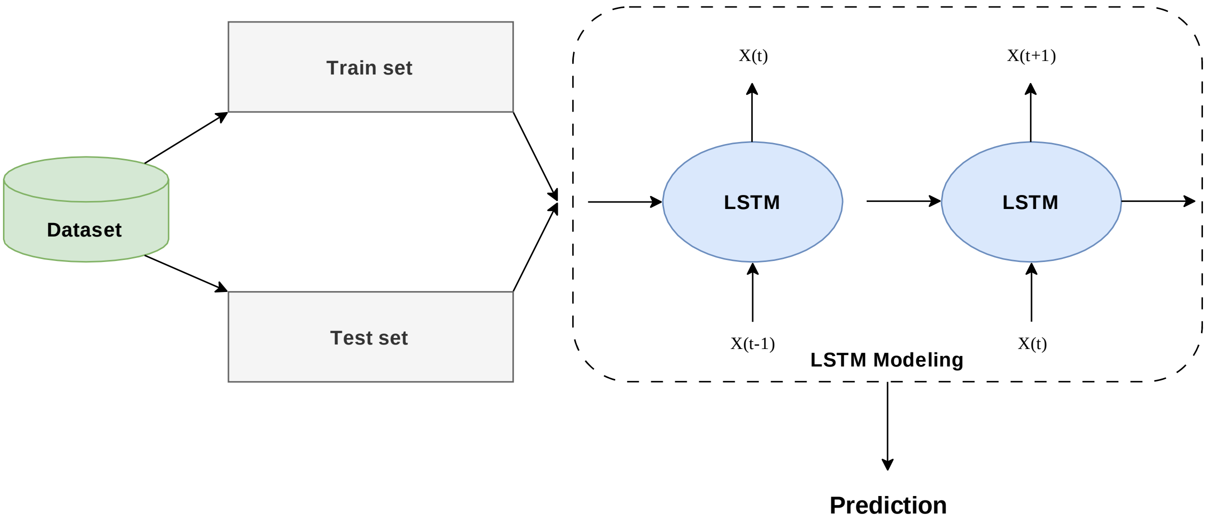


Figure 21: Architecture of the proposed model

Given the diverse range of input features for our load forecasting task, we employ suitable neural network components to acquire deep representations and extract comprehensive features from each specific input. To accomplish this, we take advantage of the rapid advancements in Deep Learning and utilize the flexible and user-friendly Python Neural Network Toolboxes.

Conclusion

In this chapter, we have introduced our smart grid model in both distribution and consumption level by giving the related equations that model the behaviour of smart electrical grid. In addition, we have proposed our AI-based solution that is based on time series forecasting for electrical load in distribution level and electricity price in consumption level. After the execution of multiple experimentations, we have identified LSTM as the best choice fore time-series forecasting in RNN models. For this reason, we have applied LSTM model to perform our time-series forecasting.

4 Implementation and Results of the proposed AI-based solution

Introduction

In this chapter, we present the tools and the results of the developed load and electricity price forecasting that are time-series forecasting.

Load forecasting using Neural Network Toolbox

1. Used tools and programming languages :

   (a) Language of Python :

   Python is a high-level programming language that is widely used for various applications, including data analysis, scientific computing, web development, and artificial intelligence. It emphasizes code readability and simplicity, making it accessible for both beginners and experienced programmers. Python supports a wide range of libraries and frameworks, making it a popular choice for diverse tasks.

   Python is a popular programming language for load forecasting due to its robust data analysis capabilities, extensive libraries for machine learning and statistical modeling, flexibility in integrating different components, active community support, and ease of deployment. It provides a comprehensive ecosystem that facilitates the entire load forecasting workflow, from data preprocessing to model development and deployment.

   Figure 22: Logo of python

   (b) Kaggle :

   Kaggle is an online platform and community that hosts data science competitions, provides datasets, and offers a collaborative environment for data scientists and machine learning practitioners. It was founded in 2010 and acquired by Google in 2017. Kaggle's primary objective is to foster knowledge sharing, encourage collaboration, and drive innovation in the field of data science and machine learning.

Here are some reasons why you should consider using Kaggle for programming in Python:

Datasets and Challenges: Kaggle provides a vast collection of datasets across various domains, including finance, healthcare, image recognition, natural language processing, and more. These datasets are openly available, and many are accompanied by specific problem statements or challenges. Working on Kaggle challenges allows you to gain hands-on experience with real-world data and problem-solving in Python.

Learning and Collaboration: Kaggle offers a collaborative environment where you can engage with a vibrant community of data scientists, machine learning enthusiasts, and experts. You can participate in discussions, ask questions, and learn from others' approaches and solutions. This collaborative aspect provides valuable insights, promotes knowledge sharing, and helps you enhance your programming skills in Python.

Code Sharing and Reproducibility: Kaggle enables you to share your code, notebooks, and solutions with others. This allows you to showcase your work, receive feedback, and learn from alternative approaches. Moreover, by sharing your code, you contribute to the reproducibility of research and encourage transparency in the field of data science.

Competitions and Rankings: Kaggle hosts data science competitions where individuals or teams can participate to solve specific challenges. Competing in these competitions helps you sharpen your skills, explore advanced techniques, and measure your performance against other participants. The rankings and leaderboard system provide motivation and recognition for your efforts.

Resources and Tutorials: Kaggle offers a wealth of resources, including tutorials, code examples, and documentation. These resources cover a wide range of topics, from beginner-level introductions to advanced machine learning algorithms and techniques. You can leverage these materials to learn new concepts, explore different Python libraries, and deepen your understanding of data science and programming.

In summary, Kaggle offers a platform for data scientists and machine learning practitioners to collaborate, learn, and compete. It provides access to diverse datasets, facilitates code sharing and collaboration, offers resources and tutorials, and promotes skill development in Python programming. Whether you are a beginner looking to learn data science or an experienced practitioner seeking to solve complex problems, Kaggle can be a valuable tool for honing your programming skills and advancing your knowledge in Python.

Figure 23: Logo of kaggle

(c) Libraries used :

We have used a set of libraries as needed in this implementation, each one played its role according to its competence, as shown in the following here:

i. Pandas :

Pandas simplifies many common data manipulation tasks, allowing you to efficiently clean, transform, and analyze datasets. It is particularly well-suited for tasks like exploratory data analysis, data preprocessing, feature engineering, and data wrangling before feeding the data into machine learning models. Pandas' intuitive and expressive syntax, along with its vast array of functionalities, has made it one of the most widely used libraries in the data science community.

ii. NumPy :

NumPy, short for Numerical Python, is a fundamental open-source Python library for scientific computing and numerical operations. It provides support for large, multi-dimensional arrays and matrices, along with a comprehensive collection of mathematical functions to operate on these arrays efficiently. NumPy serves as a foundational library for many other scientific and data analysis libraries in the Python ecosystem.

Figure 24: Logo of pandas

NumPy is a powerful Python library for efficient manipulation and computation with large arrays and matrices. Its core ndarray object, along with its extensive collection of mathematical functions, makes it an essential tool for scientific computing, data analysis, and machine learning. NumPy's performance, memory efficiency, and seamless integration with other scientific libraries contribute to its widespread use and popularity among the Python community.

Figure 25: Logo of numpy

iii. Matplotlib :

Matplotlib is a widely-used Python library for creating high-quality visualizations and plots. It provides a flexible and comprehensive set of tools for generating various types of charts, graphs, and visual representations of data. Matplotlib aims to make it easy to create publication-quality graphics, suitable for both interactive exploration and static presentation.

Matplotlib is a powerful and flexible Python library for creating high-quality visualizations and plots. It provides a wide range of plotting functions, customizable options, and an object-oriented interface for creating diverse visual representations of data. Whether you need basic line plots or complex interactive visualizations, Matplotlib offers the tools and flexibility to meet your data visualization needs.

Figure 26: Logo of matpoltip

iv. Keras :

Keras is an open-source deep learning framework written in Python. It provides a user-friendly and high-level interface to build, train, and deploy neural networks. Keras is designed to be intuitive, modular, and efficient, allowing both beginners and experienced researchers to experiment with and implement deep learning models quickly.

Keras is a user-friendly and flexible deep learning framework that simplifies the process of building and training neural networks. It provides a high-level API, modularity, and support for different backends, making it accessible to both beginners and advanced users. With its intuitive interface and extensive documentation, Keras has become a popular choice for rapid prototyping, research, and development of deep learning models.

Figure 27: Logo of keras

v. sklearn :

Scikit-learn, also known as sklearn, is a popular open-source machine learning library for Python. It provides a comprehensive set of tools and functionalities for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, and model selection. Scikit-learn is built on top of other scientific Python libraries such

Figure 28: Logo of sklearn

as NumPy, SciPy, and Matplotlib, and is designed to be simple, efficient, and accessible.

(d) Programming Environment :

Load forecasting is the process of predicting the electricity demand or load for a given time period in the future. It is an important task for utilities and power system operators to efficiently plan and manage their electricity generation and distribution. Here are the general steps involved in load forecasting:

Data Collection:

We used the dataset of Great Britain From the following website ....National Grid ESO is the electricity system operator for Great Britain. They have gathered information on the electricity demand in Great Britain since 2009. The dataset is updated twice an hour, resulting in 48 entries per day. This makes the dataset ideal for time series forecasting. The columns in the dataset are as follows:

SET T LEMET DAT A: date in the format dd/mm/yyyy.

SET T LEMENT PERIOD: half-hourly period for the historic outage occurred. ND (National Demand): National Demand is the sum of metered generation, excluding generation required to meet station load, pump storage pumping, and interconnector exports. National Demand is calculated based on National Grid ESO operational generation metering. Measured in MW. TSD (Transmission System Demand): Transmission System Demand is equal to ND plus the additional generation required to meet station load, pump storage pumping, and interconnector exports. Measured in MW.

ENGLAND W ALES DEMAND: England and Wales Demand, similar to ND but on an England and Wales basis. Measured in MW.

EMBEDDED WIND GENERAT ION: This is an estimate of the GB wind generation from wind farms that do not have Transmission System metering installed. These wind farms are embedded in the distribution network and invisible to National Grid ESO. Their e↵ect is to suppress electricity demand during periods of high wind. The true output of these generators is not known, so an estimate is provided based on National Grid ESO's best model. Measured in MW.

EMBEDDED WIND CAPACIT Y : This represents National Grid ESO's best view of the installed embedded wind capacity in GB. It is based on publicly available information compiled from various sources and is not the definitive view. The capacity estimation is consistent with the generation estimate mentioned above measured in MW.

EMBEDDED SOLAR GENERAT ION: This is an estimate of the GB solar generation from PV panels. Similar to embedded wind generation, these solar panels are embedded in the distribution network and invisible to National Grid ESO. Their e↵ect is to suppress electricity demand during periods of high radiation. The true output of these generators is not known, so an estimate is provided based on National Grid ESO's best model. Measured in MW.
EMBEDDED SOLAR CAPACIT Y : Similar to embedded wind capacity, this column represents the installed embedded solar capacity in GB. Measured in MW.
NON BM ST OR (Non-Balancing Mechanism Short-Term Operating Reserve): For units not included in the ND generator definition, this column represents generation or demand reduction. Measured in MW.
PUMP ST ORAGE PUMPING: The demand due to pumping at hydro pump storage units. The negative sign signifies pumping load.
IFA F LOW (IFA Interconnector Flow): The flow on the respective interconnector. The negative sign signifies export power out from GB, while the positive sign signifies import power into GB. Measured in MW.
IFA2 F LOW (IFA Interconnector Flow): The flow on the respective interconnector. The negative sign signifies export power out from GB, while the positive sign signifies import power into GB. Measured in MW.
MOY LE F LOW (Moyle Interconnector Flow): The flow on the respective interconnector. The negative sign signifies export power out from GB, while the positive sign signifies import power into GB. Measured in MW.
EAST WEST F LOW (East West Interconnector Flow): The flow on the respective interconnector. The negative sign signifies export power out from GB, while the positive sign signifies import power into GB. Measured in MW.
NEMO F LOW (Nemo Interconnector Flow): The flow on the respective interconnector. The negative sign signifies export power out from GB, while the positive sign signifies import power into GB. Measured in MW.
NSL F LOW (North Sea Link Interconnector Flow): The flow on the respective interconnector. The negative sign signifies export power out from GB, while the positive sign signifies import power into GB. Measured in MW.
ELCLINK F LOW : Currently not specified."

Loading of Data set :

Here we load the dataset of electricity consumption of UK 2009-2023 into kaggle figure 30 notebook using Pandas library as shows figure 29.

```
import pandas as pd
```

Figure 29: Pandas Import

```
df = pd.read_csv("/kaggle/input/electricity-consumption-uk-20092022/historic_demand_2009_2023.csv", index_col=0)

# Change column names to lower case and drop id (row number)
df.columns = df.columns.str.lower()
```

Figure 30: Loading dataset to kaggle

Data Preprocessing:

In this step, the collected data will be cleaned to eliminate any inconsistencies, errors, or outliers that may have a negative impact on the forecasting model. Various tasks, including data interpolation, imputation, and normalization, will be performed to ensure data quality and consistency. so we run this code to see the all columns as shows figure 31 : The first part of dataset is shown in figure 32 The second part of dataset is shown

```
df.sample(n=7)
```

Figure 31: DataFrame : All Columns

	settlement_date	settlement_period	nd	tsd	england_wales_demand	embedded_wind_generation	embedded_wind_capacity	embedded_solar_generation	embedded_solar_capacity	non_bm_stor	pump_stora
114819	2015-07-19	24	26064	26766	23382	1732	3897	5140	8741	0	
18494	2010-01-21	15	45155	47338	40844	491	1786	0	0	0	
6967	2009-05-25	13	22283	23715	19721	146	1673	0	0	0	
84195	2013-10-20	15	24544	25581	22101	673	2337	0	3069	0	
66431	2012-10-14	45	32287	33052	29223	216	2085	0	1937	0	
98352	2014-08-09	35	29041	29936	26614	1061	3344	1640	5107	0	
47673	2011-09-19	20	39079	40040	35374	526	1836	63	668	0	

Figure 32: Results : part I

in figure 33

pump_storage_pumping	ifa_flow	ifa2_flow	britned_flow	moyle_flow	east_west_flow	nemo_flow	nsl_flow	eleclink_flow	is_holiday
8	1900	0	952	-87	-107	0	NaN	NaN	0
12	-1426	0	0	-145	0	0	NaN	NaN	0
862	1029	0	0	-70	0	0	NaN	NaN	1
73	849	0	544	-219	-245	0	NaN	NaN	0
10	-4	0	994	-251	0	0	NaN	NaN	0
8	1996	0	1007	0	-387	0	NaN	NaN	0
9	173	0	-452	0	0	0	NaN	NaN	0

Figure 33: Results : part II

so this code rearranges the rows of the DataFrame so that they are in ascending order based on the values in these columns "settlement_date" and "settlement_period" as shows figure 34.

```
# Sort values by date
df.sort_values(
    by=["settlement_date", "settlement_period"], inplace=True, ignore_index=True
)
```

Figure 34: Sorting Code

The code df.isna().any() checks for missing values (NaN) in each column of the DataFrame df and returns a Boolean Series indicating whether each column contains any missing values. This code is given in figure 35 The code filters the DataFrame df to include

```
[10]: df.isna().any()
```

```
[10]: settlement_date              False
      settlement_period            False
      nd                           False
      tsd                          False
      england_wales_demand         False
      embedded_wind_generation     False
      embedded_wind_capacity       False
      embedded_solar_generation    False
      embedded_solar_capacity      False
      non_bm_stor                  False
      pump_storage_pumping         False
      ifa_flow                     False
      ifa2_flow                    False
      britned_flow                 False
      moyle_flow                   False
      east_west_flow               False
      nemo_flow                    False
      nsl_flow                      True
      eleclink_flow                 True
      is_holiday                   False
      dtype: bool
```

Figure 35: Nan check

only the rows where either the "eleclinkflow" or "nslflow" column contains missing values (NaN) in figure.36 and figure.37 .

```
df.loc[(df["eleclink_flow"].isna()) | (df["nsl_flow"].isna()), :]
```

	settlement_date	settlement_period	nd	tsd	england_wales_demand	embedded_wind_generation	embedded_wind_capacity	embedded_solar_generation	embedded_solar_capacity	non_bm_stor	pump_stora
0	2009-01-01	1	37910	38704	33939	54	1403	0	0	0	
1	2009-01-01	2	38047	38964	34072	53	1403	0	0	0	
2	2009-01-01	3	37380	38651	33615	53	1403	0	0	0	
3	2009-01-01	4	36426	37775	32526	50	1403	0	0	0	
4	2009-01-01	5	35687	37298	31877	50	1403	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	
175291	2018-12-31	44	26826	28428	25195	2734	5918	0	13052	0	
175292	2018-12-31	45	25660	27542	24145	2730	5918	0	13052	0	
175293	2018-12-31	46	25047	26971	23496	2726	5918	0	13052	0	
175294	2018-12-31	47	24188	26224	22683	2673	5918	0	13052	0	
175295	2018-12-31	48	23800	25785	22367	2620	5918	0	13052	0	

175296 rows × 20 columns

Figure 36: Filtering DataFrame : Results part I

This code drops the specified columns, removes rows where the "settlementperiod" value is greater than 48, and resets the index of the DataFrame fig 38.

In this code we want to Verify that both countries Wales and England have the same holidays fig 39:

Having seen that the bank holidays are the same, We can proceed with this python package to extract the bank holidays and store them in the right format.

It's worth noting that this package includes the original bank holiday and when it was observed. We will only store the observed days as shows figure 40.

```
df.loc[(df["eleclink_flow"].isna()) | (df["nsl_flow"].isna()), :]
```

d_wind_capacity	embedded_solar_generation	embedded_solar_capacity	non_bm_stor	pump_storage_pumping	ifa_flow	ifa2_flow	britned_flow	moyle_flow	east_west_flow	nemo_flow	nsl_flow	eleclink_flow	is_holiday
1403	0	0	0	33	2002	0	0	-161	0	0	NaN	NaN	1
1403	0	0	0	157	2002	0	0	-160	0	0	NaN	NaN	1
1403	0	0	0	511	2002	0	0	-160	0	0	NaN	NaN	1
1403	0	0	0	589	1772	0	0	-160	0	0	NaN	NaN	1
1403	0	0	0	851	1753	0	0	-160	0	0	NaN	NaN	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
5918	0	13052	0	13	1960	0	777	-454	-535	0	NaN	NaN	0
5918	0	13052	0	23	1182	0	-268	-455	-536	0	NaN	NaN	0
5918	0	13052	0	36	1157	0	-297	-455	-536	0	NaN	NaN	0
5918	0	13052	0	49	1546	0	-467	-454	-466	0	NaN	NaN	0
5918	0	13052	0	97	1553	0	-453	-454	-381	0	NaN	NaN	0

Figure 37: Filtering DataFrame : Results part II

```
df.drop(columns=["nsl_flow", "eleclink_flow"], axis=1, inplace=True)

# Drop rows where settlement_period value is greater than 48
df.drop(index=df[df["settlement_period"] > 48].index, inplace=True)

df.reset_index(drop=True, inplace=True)
```

Figure 38: Settlement Period Filter Code

```
# Compare England's and Wales' bank holiday
bank_holiday_england = holidays.UK(
    subdiv="England", years=range(2009, 2024), observed=True
).items()
bank_holiday_wales = holidays.UK(
    subdiv="Wales", years=range(2009, 2024), observed=True
).items()

print(bank_holiday_england == bank_holiday_wales)
```

```
True
```

Figure 39: Holiday Verification

```
# Create empty lists to store data
holiday_names = []
holiday_dates = []
holiday_names_observed = []
holiday_dates_observed = []

for date, name in sorted(bank_holiday_england):
    holiday_dates.append(date)
    holiday_names.append(name)
    # Pop the previous value as observed bank holidays takes place later
    if "Observed" in name:
        holiday_dates_observed.pop()
        holiday_names_observed.pop()

    holiday_names_observed.append(name)
    holiday_dates_observed.append(np.datetime64(date))

holiday_dates_observed[:5]
```

```
[numpy.datetime64('2009-01-01'),
 numpy.datetime64('2009-04-10'),
 numpy.datetime64('2009-04-13'),
 numpy.datetime64('2009-05-04'),
 numpy.datetime64('2009-05-25')]
```

Figure 40: Holidayes Observed Check

Once We've verified that the holidays are correctly loaded, one can compare the holiday dates variable and the date in the dataset and store the boolean output in a new column as shows figure 41 :

```
df["is_holiday"] = df["settlement_date"].apply(
    lambda x: pd.to_datetime(x) in holiday_dates_observed
)

df["is_holiday"] = df["is_holiday"].astype(int)

df[df["is_holiday"] == 1].sample(7)
```

id_generation	embedded_wind_capacity	embedded_solar_generation	embedded_solar_capacity	non_bm_stor	pump_storage_pumping	ifa_flow	ifa2_flow	britned_flow	moyle_flow	east_west_flow	nemo_flow	is_holiday
1285	1786	71	155	0	10	590	0	-461	-413	0	0	1
1547	6465	1080	13080	0	9	1959	0	979	69	124	980	1
2344	5754	0	12916	0	1620	1304	0	361	207	432	0	1
1234	1755	0	0	0	2	-1523	0	0	-412	0	0	1
450	2113	0	2585	0	490	994	0	915	-121	0	0	1
203	2313	412	2933	0	8	1068	0	458	0	-302	0	1
952	4210	0	9300	0	1409	1797	0	843	178	286	0	1

Figure 41: Holiday check

The first step in the feature creation is to change the date format to include the hourly values. The settlement period values refer to how many samples have been taken per day. Given that there are 48 samples per day, each sample represents 30 minutes of the day. Using this information together with the valuable knowledge of the StackOverflow contributors, we managed to change the date format in figure.42:

```
# Apply lambda function to turn settlement period into hours.
# Settlement period equal to 0 corresponds with 00:00:00 and each
# settlement period adds 30 minutes until settlement period 48, which
# corresponds with 23:30:00
df["period_hour"] = (df["settlement_period"]).apply(
    lambda x: str(datetime.timedelta(hours=(x - 1) * 0.5))
)

df.loc[df["period_hour"] == "1 day, 0:00:00", "period_hour"] = "0:00:00"

# Move the new column
column_to_move = df.pop("period_hour")
df.insert(2, "period_hour", column_to_move)

df.sample(n=7)
```

	settlement_date	settlement_period	period_hour	nd	tsd	england_wales_demand	embedded_wind_generation	embedded_wind_capacity	embedded_solar_generation	embedded_solar_capacity	non_bm_stor	pump_storage_pumping	ifa_flow	ifa2_flow	brit
156729	2017-12-25	28	13:30:00	33644	34840	30188	2425	5754	273	12916	0	5	1996	0	
120392	2015-11-29	23	11:00:00	35682	36130	32082	3232	4013	574	9056	156	4	1493	0	
25819	2010-06-24	48	23:30:00	28755	30071	26120	118	2056	0	30	0	816	1821	0	
132494	2016-08-07	31	15:00:00	19791	20297	18055	2998	4268	6140	10500	0	6	1406	0	
6082	2009-05-07	37	18:00:00	39277	39927	35678	1367	1673	0	0	0	0	1855	0	
230376	2022-03-09	3	1:00:00	24174	26588	22651	2555	6545	0	13738	0	941	-221	-652	
134070	2016-09-09	23	11:00:00	31131	31712	28669	2476	4403	3560	11433	0	4	-77	0	

Figure 42: Change the Date Format .

As can be seen, a new column called periodhour includes the hour at which the electricity demand measurement was taken. One can now combine it with the actual date as follows in figure.43:

Feature Selection:

So in this step the collected data will be analyzed to identify relevant features or variables that have a significant impact on electricity load. This analysis aims to uncover factors such as time of day, day of the week, holidays, weather conditions, and other location-specific or context-specific factors that play a crucial role in load forecasting. By identifying these features, it becomes possible to incorporate them into the forecasting model to improve its accuracy and reliability.

```
df["settlement_date"] = pd.to_datetime(
    (df["settlement_date"] + " " + df["period_hour"])
)

df.sample(n=5)
```

	settlement_date	settlement_period	period_hour	nd	tsd	england_wales_demand	embedded_wind_generation	embedded_wind_capacity	embedded_solar_generation	embedded_solar_capacity	non_bm_stor	pump_storage_pumping	ifa_flow	ifa2_flow	brit
205663	2020-10-10 03:30:00	8	3:30:00	20144	21641	18428	1707	6527	0	13080	0	997	199	0	
23084	2010-04-29 00:00:00	1	0:00:00	26419	28286	23680	1015	1883	0	21	0	1341	921	0	
68987	2012-12-23 09:30:00	20	9:30:00	34069	35124	30856	848	2085	53	2026	0	7	994	0	
186971	2019-09-16 16:30:00	34	16:30:00	33168	34672	30614	1642	6182	1890	13080	0	14	1438	0	
113449	2015-07-07 19:30:00	40	19:30:00	32013	32615	29357	1887	3897	733	8719	0	10	1998	0	

Figure 43: Check column Period Hour .

We will also explore the distribution of electricity demand with respect the di↵erent features, such as as hour, month or year. This is a great way to understand the seasonalities in the time series: so in this plot it is show the Distribution of electricity consumption with hours We can also notice the hours in which it consumes more energy 44:

```
fig, ax = plt.subplots(figsize=(15, 5))
sns.boxplot(x="settlement_period", y="tsd", data=df)
# Reduce the frequency of the xticks and change the labels to be in
# the range [0,24] hours
ax.set_xticks(range(1, 49, 2))
ax.set_xticklabels(range(1, 25))
ax.set_xlabel("Hour")
ax.set_ylabel("Electricity demand (MW)")
ax.set_title("Distribution of electricity consumption with hours");
```

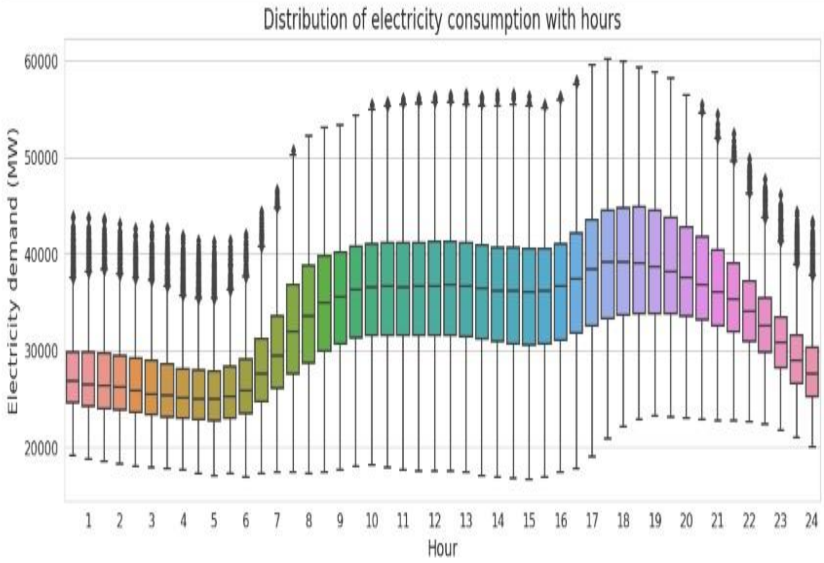


Figure 44: Distribution of electricity consumption with hours

In this plot it is show the Distribution of electricity consumption with months We can also note the months in which energy is consumed more in figure.45:
In this plot, the Distribution of electricity consumption within years is shown. We can also note the years in which it consumes more energy in figure 46.

```
fig, ax = plt.subplots(figsize=(15, 5))
sns.boxplot(x="month", y="tsd", data=df)

ax.set_xlabel("Month")
ax.set_ylabel("Electricity demand (MW)")
ax.set_title("Distribution of electricity consumption with months");
```

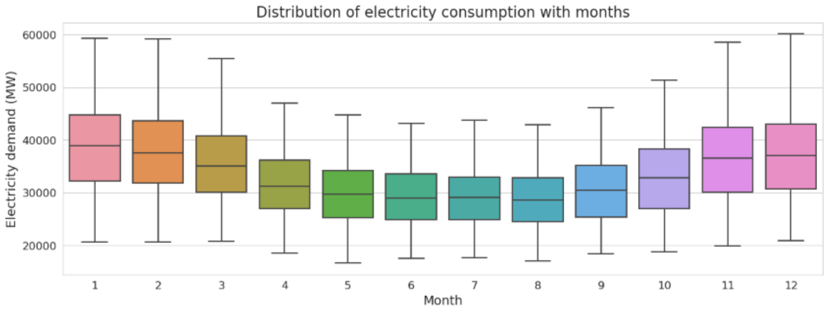


Figure 45: Distribution of electricity consumption with months

```
fig, ax = plt.subplots(figsize=(15, 5))
sns.boxplot(x="year", y="tsd", data=df)

ax.set_xlabel("Year")
ax.set_ylabel("Electricity demand (MW)")
ax.set_title("Distribution of electricity consumption with years");
```

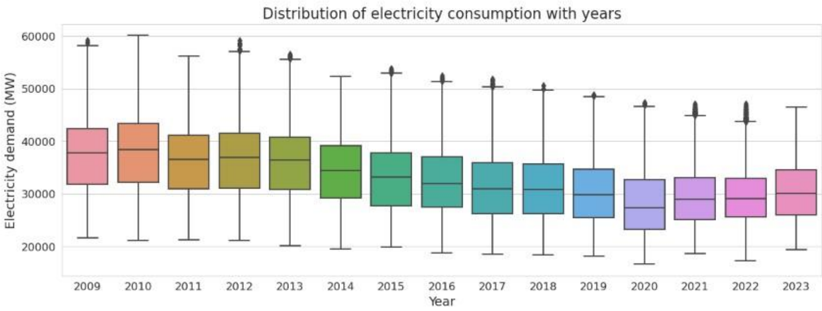


Figure 46: Distribution of electricity consumption with years

Model Selection:

So in this step we have selected three models which are LSTM ,FNN and GRU I chose these models because they are suitable for prediction based on the characteristics of the data.

LSTM

Training - test split: the first step is to split the data. I will split the data into train, test and hold-out set. The hold-out set will be used for independent evaluation of the model while the model is being trained in figure.47. In order to compare the results of the

```
threshold_date_1 = "06-01-2019"
threshold_date_2 = "06-01-2021"
train_data = df.loc[df.index < threshold_date_1]
test_data = df.loc[(df.index >= threshold_date_1) & (df.index < threshold_date_2)]
hold_out_data = df.loc[df.index >= threshold_date_2]
```

Figure 47: Training :Test Split

models, we use the Mean Absolute Percentage Error, which we implemented as follows in figure 48.

also we use root mean squared error as shown in figure.49.

```python
def mean_absolute_percentage_error(y_true, y_pred):
    """
    Calculate Mean Absolute Pertange Error given the true and
    predicted values

    Args:
        - y_true: true values
        - y_pred: predicted values

    Returns:
        - mape: MAPE value for the given predicted values
    """

    y_true, y_pred = np.array(y_true), np.array(y_pred)
    mape = np.mean(np.abs((y_true - y_pred) / y_true)) * 100
    return mape
```

Figure 48: Mean Absolute Percentage Error

```python
# Define rmse metric for keras to use as a loss function
def root_mean_squared_error(y_true, y_pred):
    return K.sqrt(K.mean(K.square(y_pred - y_true)))
```

Figure 49: Root Mean Squared Error

The code prepares the data for training and testing a Keras machine learning model. It splits the original dataset into training and testing subsets, selects specific features and the target variable, scales the data, and reshapes it into the required format for input to a Keras model in figure.50. LSTM model in figure.51.

```python
train_data = df.loc[df.index < threshold_date_1]
test_data = df.loc[(df.index >= threshold_date_1) & (df.index < threshold_date_2)]
hold_out_data = df.loc[df.index >= threshold_date_2]

# Define the features and target variable
FEATURES = [
    "is_holiday",
    "settlement_period",
    "day_of_month",
    "day_of_week",
    "day_of_year",
    "quarter",
    "month",
    "year",
    "week_of_year",
]
TARGET = "tsd"

FEATURES_TARGET = FEATURES.copy()
FEATURES_TARGET.append(TARGET)
train_data_keras = train_data[FEATURES_TARGET]
test_data_keras = test_data[FEATURES_TARGET]

scaler = MinMaxScaler(feature_range=(0,1))
train_data_keras_s = scaler.fit_transform(train_data_keras.values)
test_data_keras_s = scaler.transform(test_data_keras.values)

X_train_keras = (
    train_data_keras_s[:,:-1].
    reshape(train_data_keras_s.shape[0],1,len(FEATURES))
)
y_train_keras = train_data_keras_s[:,-1]

X_test_keras = (
    test_data_keras_s[:,:-1].
    reshape(test_data_keras_s.shape[0],1,len(FEATURES))
)
y_test_keras = test_data_keras_s[:,-1]
```

Figure 50: Keras : Define Features and Target

```python
# Define a random seed for reproducibility
tf.random.set_seed(221)

# Create and compite neural network
model = Sequential()
model.add(LSTM(256, input_shape=(X_train_keras.shape[1], X_train_keras.shape[2])))
model.add(Dropout(0.5))

model.add(Dense(1))
model.compile(loss = root_mean_squared_error, optimizer="adam")

# Define callbacks
monitor_param = root_mean_squared_error
mode="min"
early_stopping = EarlyStopping(monitor=root_mean_squared_error, patience=8, verbose=0, mode=mode)
checkpoint_save = ModelCheckpoint(
    "./models_data/simple_lstm/checkpoint",
    save_weights_only=True,
    monitor=monitor_param,
    mode=mode,
)
reduce_lr_loss = ReduceLROnPlateau(
    monitor=monitor_param, factor=0.1, patience=5, verbose=0, mode=mode
)

# Fit model
history = model.fit(
    X_train_keras,
    y_train_keras,
    epochs=100,
    batch_size=144,
    validation_data=(X_test_keras, y_test_keras),
    callbacks=[early_stopping, checkpoint_save, reduce_lr_loss]
)
```

```
Epoch 1/100
1263/1263 [==============================] - 15s 10ms/step - loss: 0.1303 - val_loss: 0.0967 - lr: 0.0010
Epoch 2/100
1263/1263 [==============================] - 13s 10ms/step - loss: 0.1041 - val_loss: 0.0903 - lr: 0.0010
Epoch 3/100
1263/1263 [==============================] - 13s 10ms/step - loss: 0.0977 - val_loss: 0.0881 - lr: 0.0010
Epoch 4/100
1263/1263 [==============================] - 12s 10ms/step - loss: 0.0932 - val_loss: 0.0883 - lr: 0.0010
Epoch 5/100
```

Figure 51: LSTM model

Figure.52 shows the LSTM loss plot.

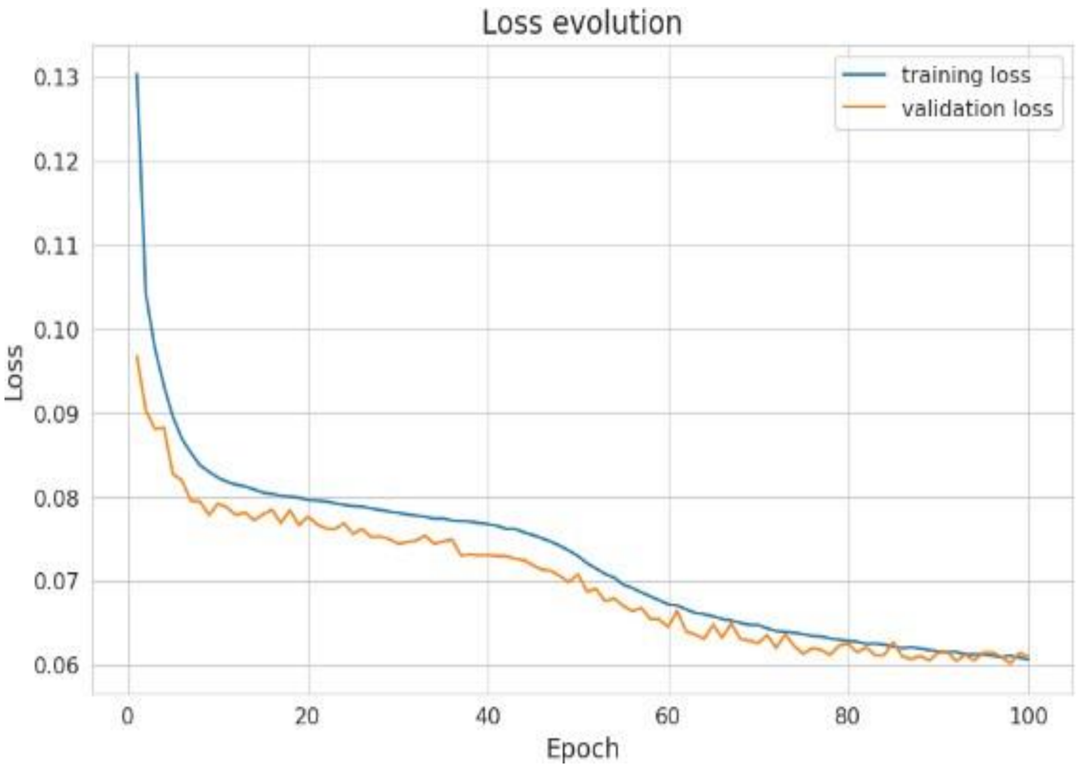


Figure 52: Plot LSTM Loss

Figure.53 shows the LSTM prediction.

```
fig, ax = plt.subplots(figsize=(15, 5))
ax.plot(result_frame.index, result_frame["tsd"], "o", label="Test set")
ax.plot(result_frame.index, result_frame["pred_lstm"], "o", label="Prediction")

ax.legend(loc="center", bbox_to_anchor=(1.075, 0.5))

ax.set_title("Prediction on test set")
ax.set_ylabel("Energy Demand (MW)")
ax.set_xlabel("Date");
```

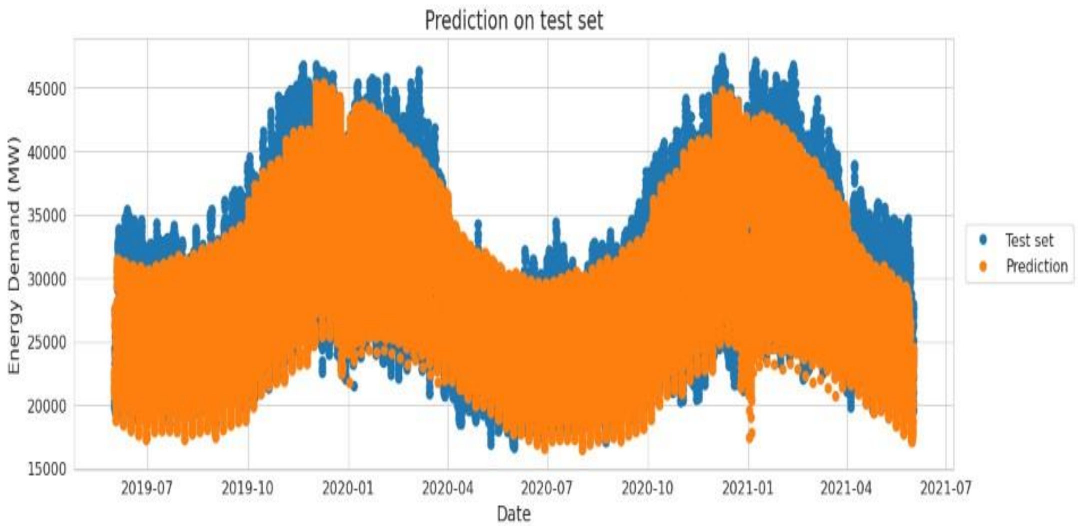


Figure 53: LSTM Prediction

Figure.54 shows the code for LSTM prediction in 2 weeks.

```
begin = "02-01-2021"
end = "02-14-2021"

fig, ax = plt.subplots(figsize=(20, 5))

ax.plot(
    result_frame.loc[(result_frame.index > begin) & (result_frame.index < end)].index,
    result_frame.loc[(result_frame.index > begin) & (result_frame.index < end)]["tsd"],
    "-o",
    label="Test set",
)

ax.plot(
    result_frame.loc[(result_frame.index > begin) & (result_frame.index < end)].index,
    result_frame.loc[(result_frame.index > begin) & (result_frame.index < end)][
        "pred_lstm"
    ],
    "-s",
    label="LSTM",
)

ax.legend(loc="center", bbox_to_anchor=(1.075, 0.5))

ax.set_title("Prediction on test set - Two weeks")
ax.set_ylabel("Energy Demand (MW)")
ax.set_xlabel("Date");
```

Figure 54: LSTM Prediction in 2 Weeks

Figure.55 shows the results for LSTM prediction in 2 weeks.

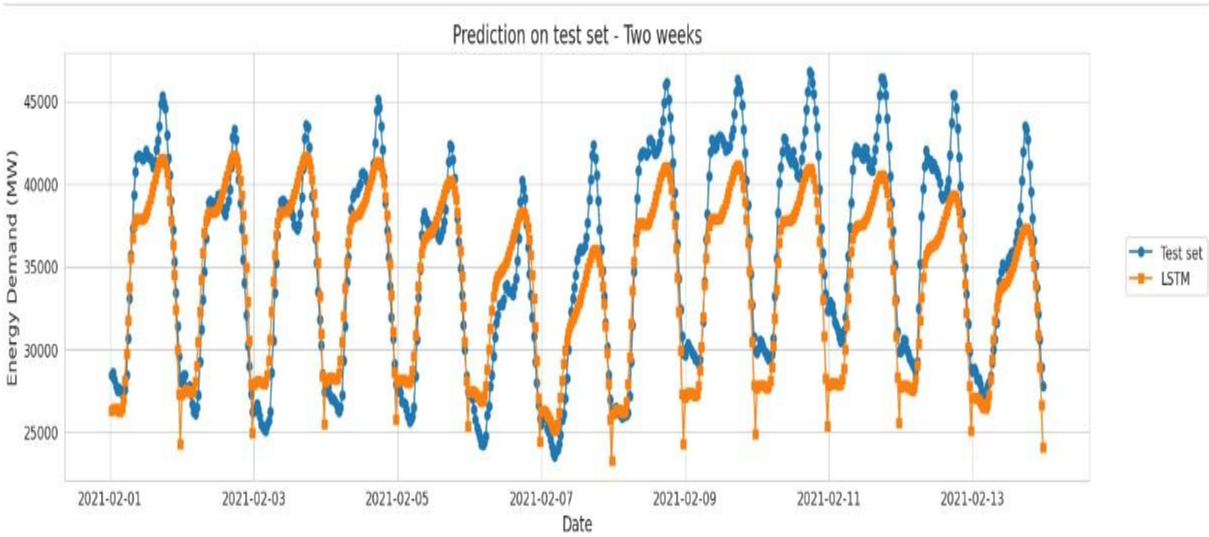


Figure 55: Result LSTM Prediction in 2 Weeks

Figure.56 shows code for mape LSTM and rmse LSTM.

```
mape_lstm = mean_absolute_percentage_error(
    y_test, result_frame["pred_lstm"]
)

rmse_lstm = np.sqrt(mean_squared_error(y_test, result_frame["pred_lstm"]))

print(
    "Mean Absolute Percentage Error of the LSTM model is: %.2f" % mape_lstm
)

print(
    "Root Mean Squared Error of the LSTM model is: %.2f MW" % rmse_lstm
)
```

```
Mean Absolute Percentage Error of the LSTM model is: 7.48
Root Mean Squared Error of the LSTM model is: 2740.58 MW
```

Figure 56: MAPE LSTM and RMSE LSTM

Feedforward Neural Network (FNN)

FNN is a type of artificial neural network where the information flows in only one direction, from the input layer to the output layer. It is one of the simplest and most common neural network architectures.

In an FNN model, the network is composed of multiple layers of interconnected artificial neurons. The first layer is the input layer, which receives the initial data or features. The subsequent layers are called hidden layers, and the final layer is the output layer, which produces the network's predictions or classifications. Each neuron in an FNN receives inputs from the neurons in the previous layer and applies a non-linear activation function to the weighted sum of its inputs. This activation function introduces non-linearity into the model, allowing it to capture complex patterns and relationships in the data. FNN models are often used for various tasks, including regression (predicting continuous values) and classification (assigning labels to data points). They have been successfully applied in a wide range of domains, such as image recognition, natural language processing, and financial forecasting.

Overall, FNNs provide a foundational building block in deep learning and serve as a basis for more complex architectures like convolutional neural networks (CNNs) and recurrent neural networks (RNNs).

Figure.57 shows the code for FNN model.

```
# Define a random seed for reproducibility
tf.random.set_seed(221)

# Create and compile neural network
model = Sequential()
model.add(Dense(256, activation='relu', input_shape=(len(FEATURES),)))
model.add(Dropout(0.5))

model.add(Dense(1))
model.compile(loss=root_mean_squared_error, optimizer="adam")

# Define callbacks
monitor_param = root_mean_squared_error
mode = "min"
early_stopping = EarlyStopping(monitor=root_mean_squared_error, patience=8, verbose=0, mode=mode)
checkpoint_save = ModelCheckpoint(
    "./models_data/simple_fnn/checkpoint",
    save_weights_only=True,
    monitor=monitor_param,
    mode=mode,
)
reduce_lr_loss = ReduceLROnPlateau(
    monitor=monitor_param, factor=0.1, patience=5, verbose=0, mode=mode
)

# Fit model
history = model.fit(
    X_train_keras,
    y_train_keras,
    epochs=100,
    batch_size=144,
    validation_data=(X_test_keras, y_test_keras),
    callbacks=[early_stopping, checkpoint_save, reduce_lr_loss]
)
```

```
Epoch 1/100
1263/1263 [==============================] - 4s 3ms/step - loss: 0.1162 - val_loss: 0.0569 - lr: 0.0010
Epoch 2/100
1263/1263 [==============================] - 4s 3ms/step - loss: 0.0847 - val_loss: 0.0466 - lr: 0.0010
Epoch 3/100
1263/1263 [==============================] - 3s 3ms/step - loss: 0.0818 - val_loss: 0.0424 - lr: 0.0010
```

Figure 57: FNN model

Figure.58 shows the plot of FNN loss.

Figure.59 shows the FNN prediction.

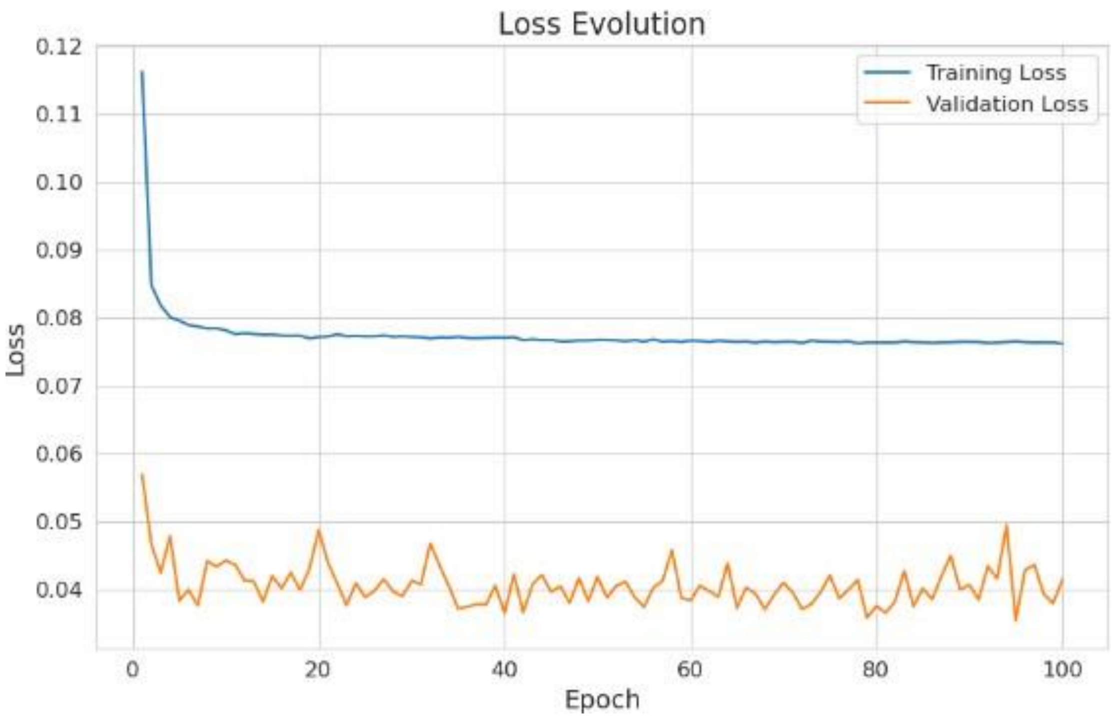


Figure 58: FNN Loss

```
fig, ax = plt.subplots(figsize=(15, 5))
ax.plot(result_frame.index, result_frame["tsd"], "o", label="Test set")
ax.plot(result_frame.index, result_frame["pred_fnn"], "o", label="Prediction")

ax.legend(loc="center", bbox_to_anchor=(1.075, 0.5))

ax.set_title("Prediction on test set")
ax.set_ylabel("Energy Demand (MW)")
ax.set_xlabel("Date")
```

```
Text(0.5, 0, 'Date')
```

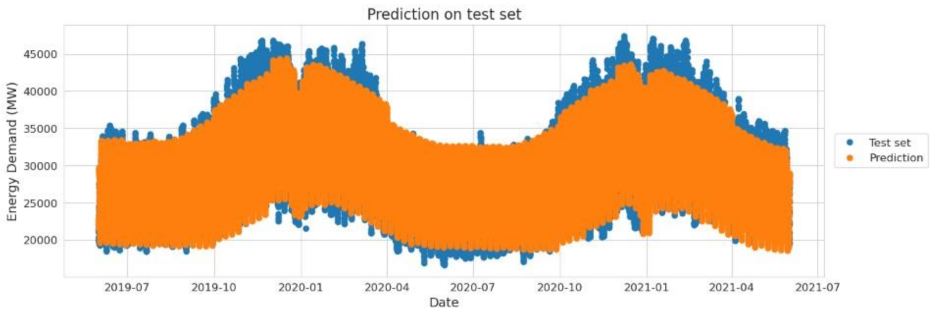


Figure 59: FNN Prediction

Figure.60 shows FNN prediction in 2 weeks.
Figure.61 shows the code for mape FNN and rmse FNN.

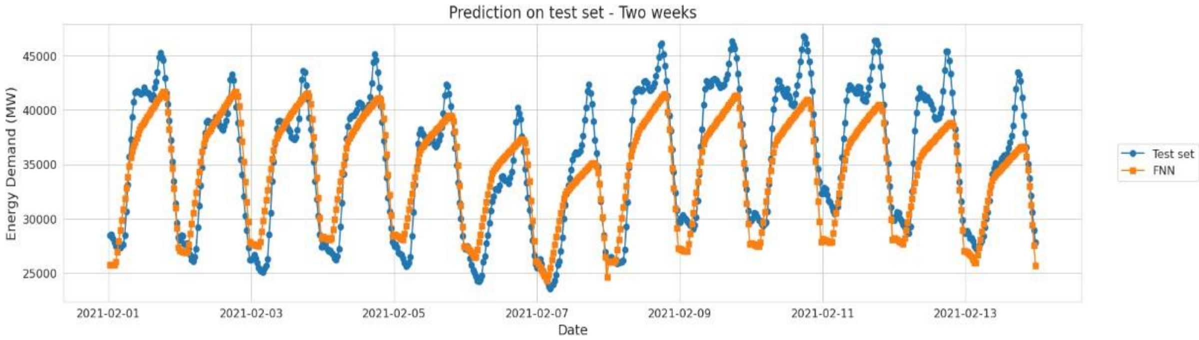


Figure 60: FNN Prediction in 2 Weeks

```
mape_fnn = mean_absolute_percentage_error(
    y_test, result_frame["pred_fnn"]
)

rmse_fnn = np.sqrt(mean_squared_error(y_test, result_frame["pred_fnn"]))

print(
    "Mean Absolute Percentage Error of the FNN model is: %.2f" % mape_fnn
)

print(
    "Root Mean Squared Error of the FNN model is: %.2f MW" % rmse_fnn
)
```

```
Mean Absolute Percentage Error of the FNN model is: 9.18
Root Mean Squared Error of the FNN model is: 3182.49 MW
```

Figure 61: MAPE FNN and RSME FNN

Gated Recurrent Unit (GRU)

GRU stands for , which is a type of recurrent neural network (RNN) architecture. It is designed to address some of the limitations of traditional RNNs in capturing long-term dependencies in sequential data.

In a GRU model, the network contains recurrent units or cells that maintain an internal state or memory. This memory allows the network to retain information from past time steps and use it to make predictions at the current time step. One of the key features of a GRU cell is its ability to selectively update and reset the memory state. It achieves this through gating mechanisms that regulate the flow of information. Specifically, a GRU cell has two gates: an update gate and a reset gate.

The update gate determines how much of the previous memory should be retained and how much of the new information should be added to the memory. It controls the trade-o↵ between remembering past information and incorporating new information. The reset gate, on the other hand, decides how much of the previous memory should be ignored when computing the current state. It allows the model to selectively reset or forget information that is no longer relevant.

By adaptively updating and resetting its memory, the GRU model can e↵ectively capture long-term dependencies in sequential data while mitigating the vanishing gradient problem often encountered in traditional RNNs. GRU models have been widely used in various natural language processing (NLP) tasks, such as machine translation, sentiment analysis, and language generation. They have also been employed in other sequential data domains, including speech recognition, time series analysis, and recommendation systems. Overall, GRUs o↵er a more efficient and powerful alternative to traditional RNNs, en-

abling better modeling of sequential data and facilitating the development of more accurate and robust predictive models.
Figure.62 shows the code for GRU model.

```
# Create and compile neural network
model = Sequential()
model.add(GRU(256, input_shape=(X_train_keras.shape[1], X_train_keras.shape[2])))
model.add(Dropout(0.5))
model.add(Dense(1))
model.compile(loss=root_mean_squared_error, optimizer="adam")

# Fit model
history = model.fit(
    X_train_keras,
    y_train_keras,
    epochs=100,
    batch_size=144,
    validation_data=(X_test_keras, y_test_keras),
    callbacks=[early_stopping, checkpoint_save, reduce_lr_loss]
)

# Prediction on test set
pred_gru = model.predict(X_test_keras)

# Inverse transform the prediction
# Since the scaler was fit using all the data (9 features + outcome variable),
# we need to store the prediction in a copy of the original data
results_gru = test_data_keras_s.copy()
results_gru[:, -1] = pred_gru.reshape(pred_gru.shape[0])
results_gru = scaler.inverse_transform(results_gru)

# Store inverse transformed predictions in the result dataframe
result_frame["pred_gru"] = results_gru[:, -1]
```

```
Epoch 1/100
1263/1263 [==============================] - 13s 9ms/step - loss: 0.1332 - val_loss: 0.1004 - lr: 0.0010
Epoch 2/100
1263/1263 [==============================] - 11s 8ms/step - loss: 0.1099 - val_loss: 0.0911 - lr: 0.0010
Epoch 3/100
1263/1263 [==============================] - 11s 9ms/step - loss: 0.1007 - val_loss: 0.0894 - lr: 0.0010
Epoch 4/100
1263/1263 [==============================] - 11s 9ms/step - loss: 0.0959 - val_loss: 0.0924 - lr: 0.0010
Epoch 5/100
```

Figure 62: GRU model

Figure.63 shows the plot of GRU loss.

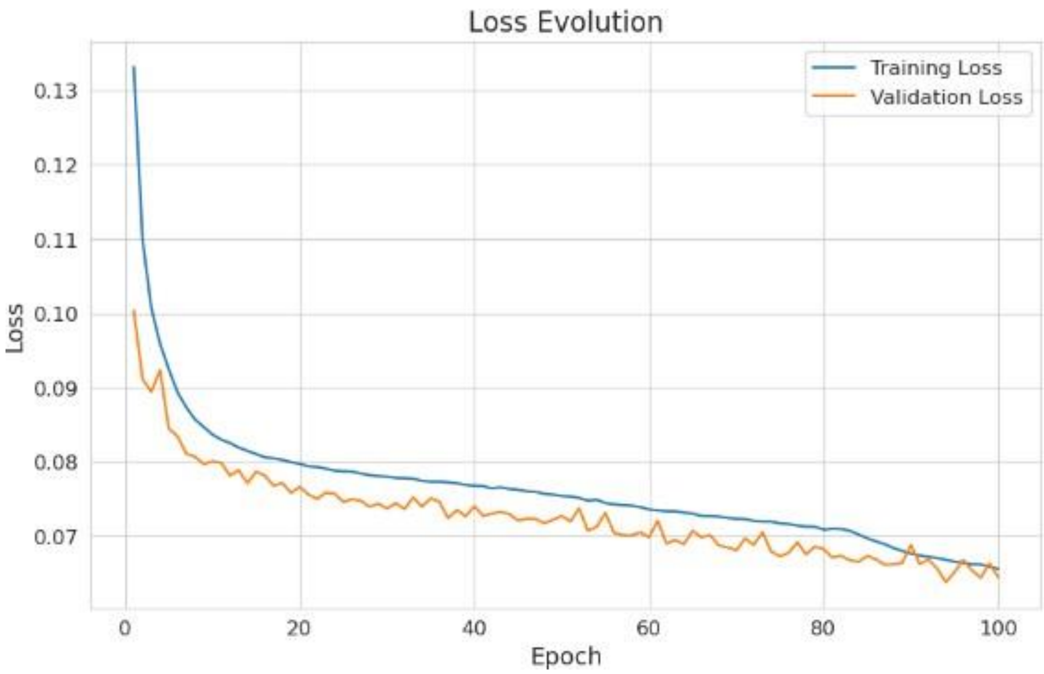


Figure 63: GRU Loss

Figure.64 shows the GRU prediction.
Figure.65 shows the code for mape GRU and rmse GRU.

```
fig, ax = plt.subplots(figsize=(15, 5))
ax.plot(result_frame.index, result_frame["tsd"], "o", label="Test set")
ax.plot(result_frame.index, result_frame["pred_gru"], "o", label="Prediction")

ax.legend(loc="center", bbox_to_anchor=(1.075, 0.5))

ax.set_title("Prediction on test set")
ax.set_ylabel("Energy Demand (MW)")
ax.set_xlabel("Date")
```

```
Text(0.5, 0, 'Date')
```

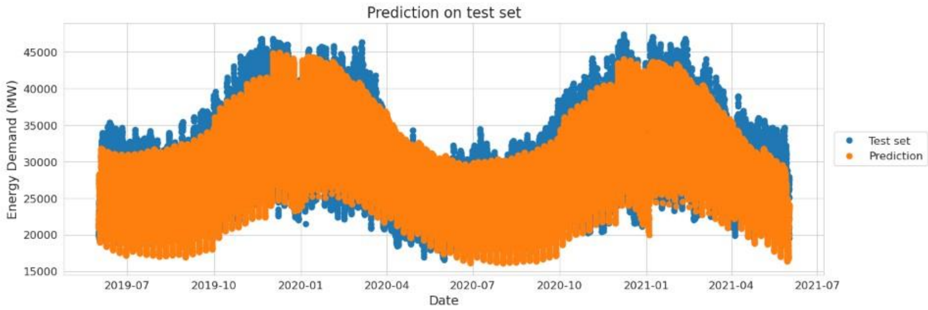


Figure 64: GRU Prediction

```
mape_gru = mean_absolute_percentage_error(y_test, result_frame["pred_gru"])
rmse_gru = np.sqrt(mean_squared_error(y_test, result_frame["pred_gru"]))

print("Mean Absolute Percentage Error of the GRU model is: %.2f" % mape_gru)
print("Root Mean Squared Error of the GRU model is: %.2f MW" % rmse_gru)
```

```
Mean Absolute Percentage Error of the GRU model is: 8.01
Root Mean Squared Error of the GRU model is: 2872.22 MW
```

Figure 65: MAPE GRU and RMSE GRU

Model Evaluation: We will Evaluate the performance of the trained model using appropriate evaluation metrics, such as root mean square error (RMSE), or mean absolute percentage error (MAPE). These metrics provide insights into the accuracy and reliability of the forecasting model in figure.66.

```
summary_df = pd.DataFrame(
    {
        "LSTM": [mape_lstm, rmse_lstm],
        "FNN": [mape_fnn, rmse_fnn],
        "GRU": [mape_gru, rmse_gru],
        "Metric": ["MAPE", "RMSE"]
    }
)

summary_df.set_index("Metric", inplace=True)

summary_df.style.format('{:.2f}')
```

	LSTM	FNN	GRU
Metric			
MAPE	7.48	9.18	8.01
RMSE	2740.58	3182.49	2872.22

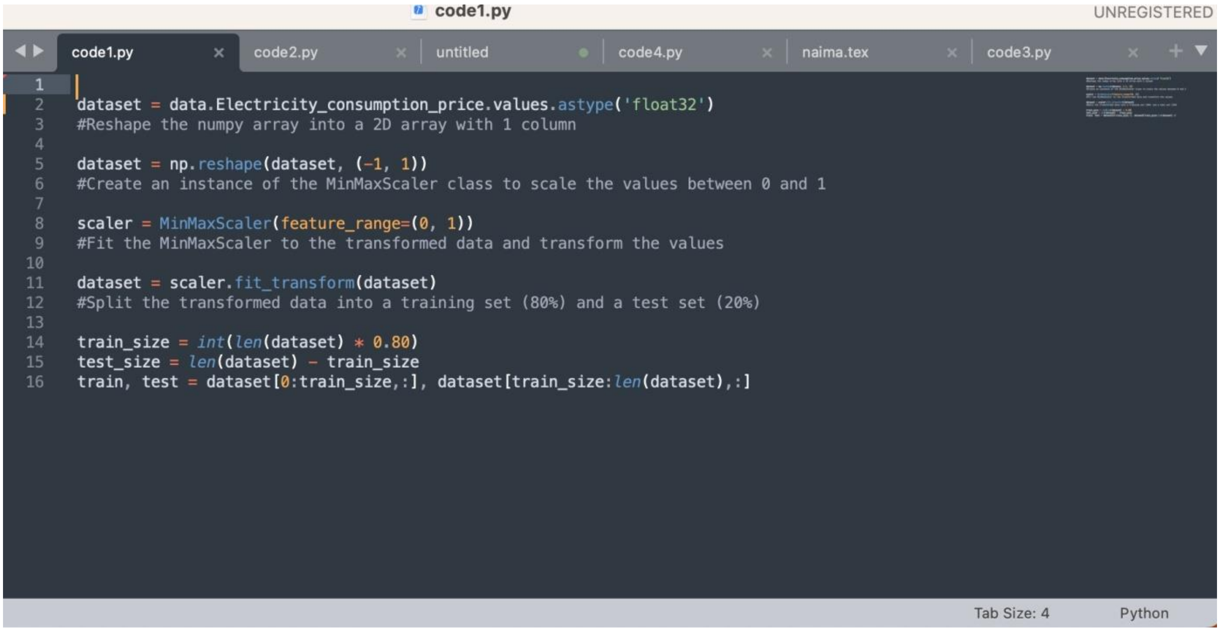
Figure 66: LSTM vs FNN vs GRU

Thus, here the best model is LSTM.

Electricity price forecasting

Modeling and evaluation

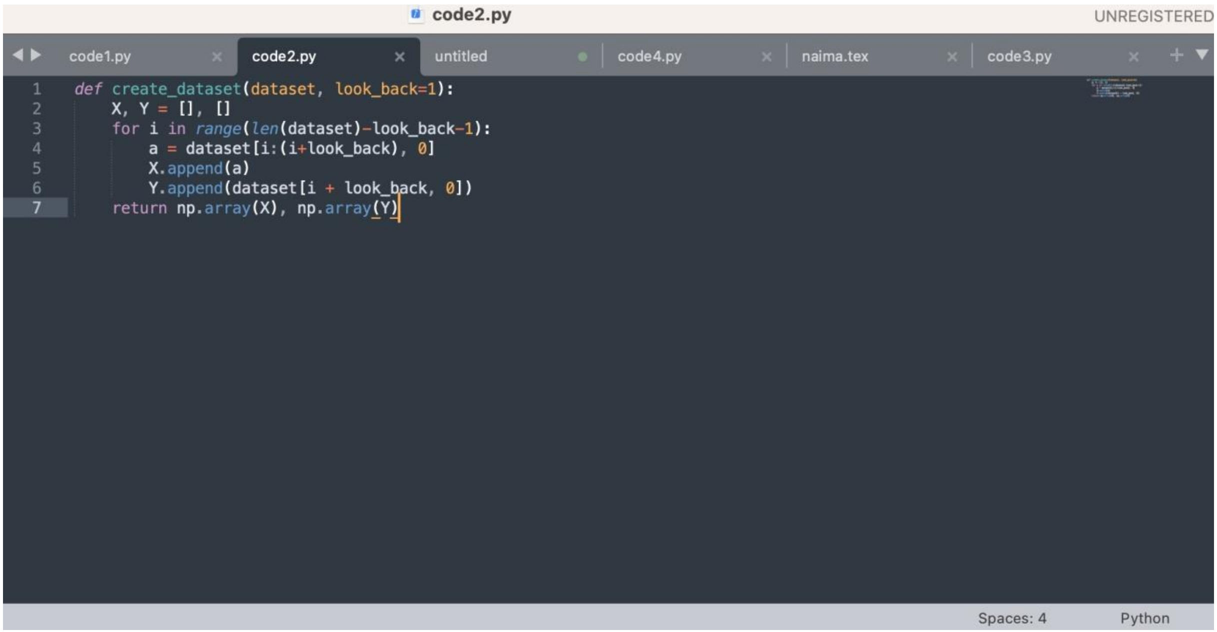
Here we transform the Electricity consumption price column of the data DataFrame into a numpy array of float values.



```python

dataset = data.Electricity_consumption_price.values.astype('float32')
#Reshape the numpy array into a 2D array with 1 column

dataset = np.reshape(dataset, (-1, 1))
#Create an instance of the MinMaxScaler class to scale the values between 0 and 1

scaler = MinMaxScaler(feature_range=(0, 1))
#Fit the MinMaxScaler to the transformed data and transform the values

dataset = scaler.fit_transform(dataset)
#Split the transformed data into a training set (80%) and a test set (20%)

train_size = int(len(dataset) * 0.80)
test_size = len(dataset) - train_size
train, test = dataset[0:train_size,:], dataset[train_size:len(dataset),:]
```

Figure 67: Transform intp array of float

Here we convert an array of values into a dataset matrix.



```python
def create_dataset(dataset, look_back=1):
    X, Y = [], []
    for i in range(len(dataset)-look_back-1):
        a = dataset[i:(i+look_back), 0]
        X.append(a)
        Y.append(dataset[i + look_back, 0])
    return np.array(X), np.array(Y)
```

Figure 68: Dataset matrix conversion

Here, we reshape into X=t and Y=t+1

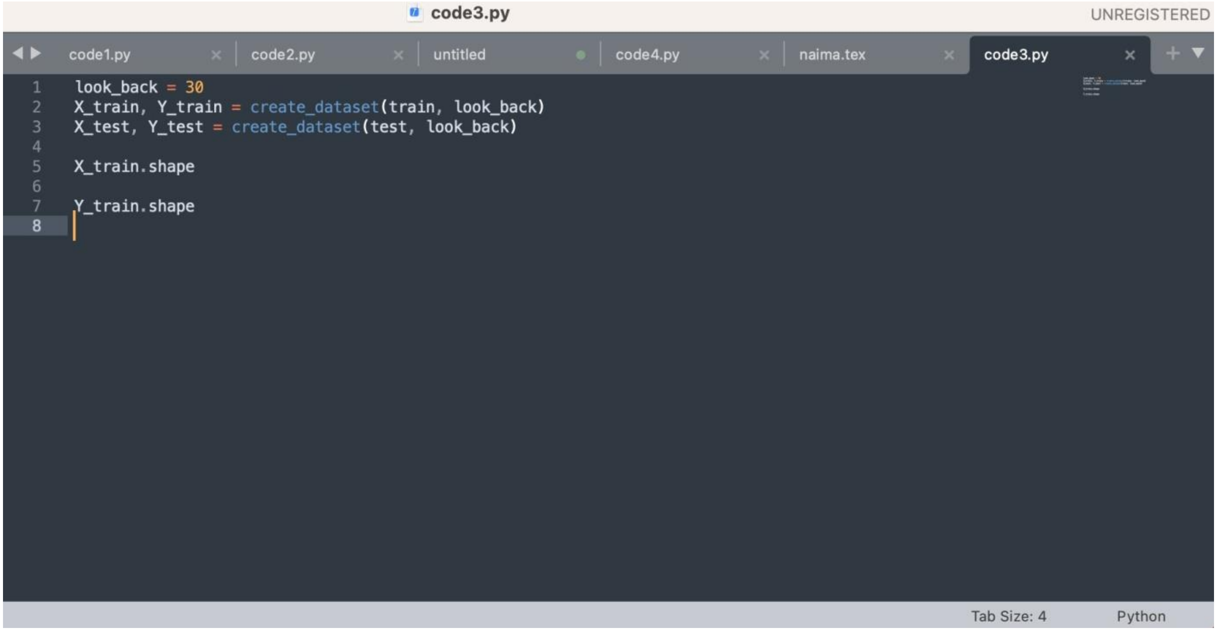


Figure 69: X and Y reshape

here we reshape input to be [samples, time steps, features]

Figure 70: Input reshape

LSTM model

Here we define our LSTM model. Adding the first layer with 365 LSTM units and input shape of the data.

```python
model = Sequential()


model.add(LSTM(365, input_shape=(X_train.shape[1], X_train.shape[2])))

# Adding a dropout layer to avoid overfitting
model.add(Dropout(0.2))

# Adding a dense layer with 1 unit to make predictions
model.add(Dense(1))

# Compiling the model with mean squared error as the loss function and using Adam optimizer
model.compile(loss='mean_squared_error', optimizer='adam')

# Fitting the model on training data and using early stopping to avoid overfitting
history = model.fit(X_train, Y_train, epochs=20, batch_size=1240, validation_data=(X_test, Y_test),
                    callbacks=[EarlyStopping(monitor='val_loss', patience=4)], verbose=1, shuffle=False)

# Displaying a summary of the model
model.summary()
```

Figure 71: Defining LSTM model

Evaluation

Here we make our predictions.

```python
train_predict = model.predict(X_train)
test_predict = model.predict(X_test)
# invert predictions
train_predict = scaler.inverse_transform(train_predict)
Y_train = scaler.inverse_transform([Y_train])
test_predict = scaler.inverse_transform(test_predict)
Y_test = scaler.inverse_transform([Y_test])

print('Train Mean Absolute Error:', mean_absolute_error(Y_train[0], train_predict[:,0]))
print('Train Root Mean Squared Error:',np.sqrt(mean_squared_error(Y_train[0], train_predict[:,0])))
print('Test Mean Absolute Error:', mean_absolute_error(Y_test[0], test_predict[:,0]))
print('Test Root Mean Squared Error:',np.sqrt(mean_squared_error(Y_test[0], test_predict[:,0])))
```

Figure 72: Predictions and evaluation

Here we have the results for model loss:
And here we have the results of the prediction.

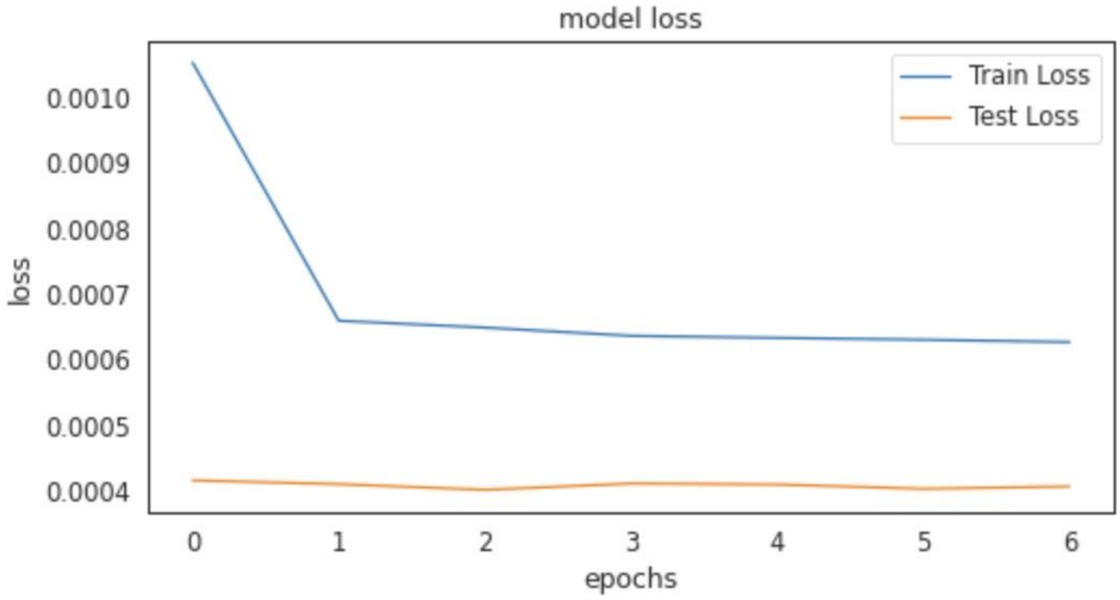


Figure 73: Model loss

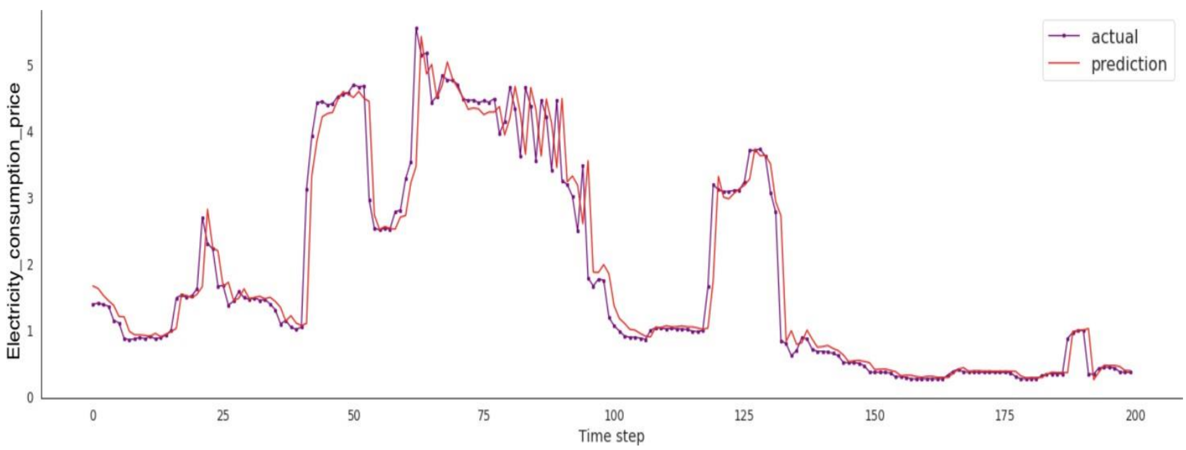


Figure 74: Prediction results

Energy Management simulation with Matpower package

To validate the results of the proposed AI-based solution, we perform some simulation tests to a real smart grid network which is the IEEE 30-bus system. This system consists of 30 nodes where we have a 6 generation nodes. This simulation is performed with Matpower package (24).

Energy generation cost results

Figure. 75 presents the results of generation cost after a load forecasting based on RNN, GRU and the proposed method which is LSTM. As shows the figure, the proposed LSTM

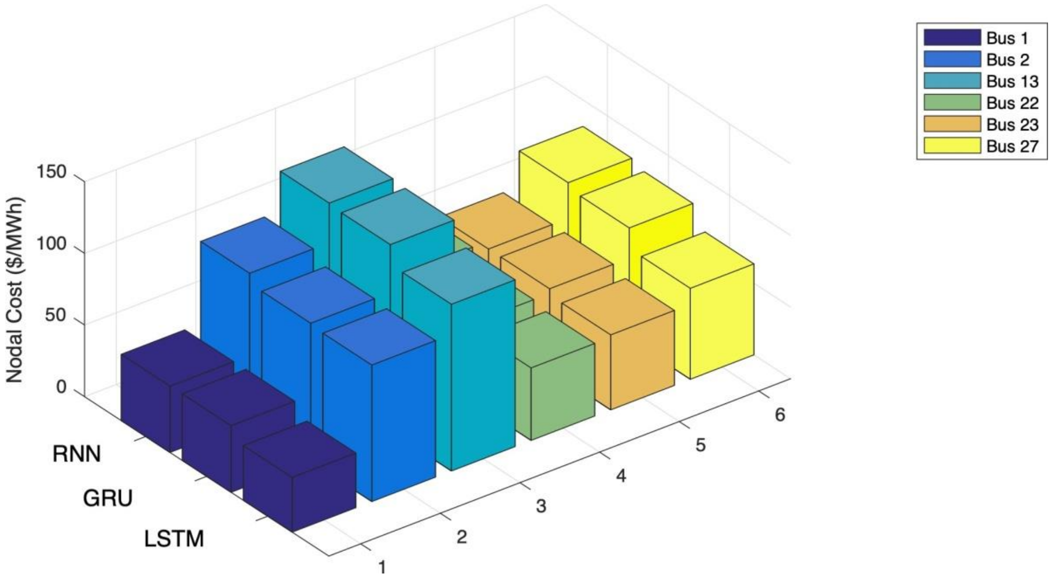


Figure 75: Energy generation cost

based load forecasting method leads to a better result in terms of energy cost which has the least cost here.

Energy consumption cost results

Figure. 76 presents the results of electricity consumption cost for four methods.

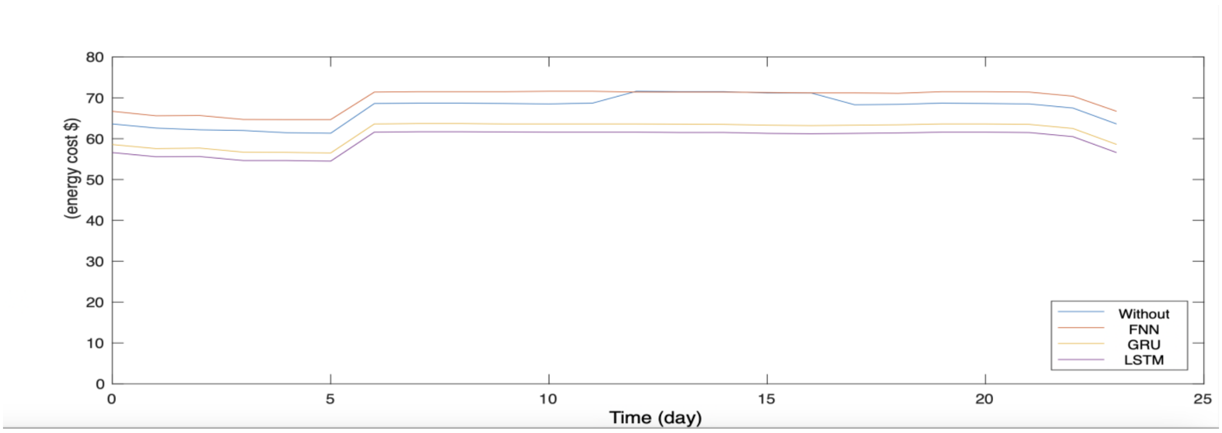


Figure 76: Electricity consumption cost

Thes four methods are as follows : without performing electricity price forecast, after an electricity price forecasting based on RNN, GRU and the proposed method which is LSTM. As shown in figure.76, the proposed LSTM based electricity price forecasting leads to a significant reduction in energy consumption cost when comparing with the other methods.

Conclusion

In this chapter, we presented the results of the load time-series forecasting in distribution level with the proposed LSTM method and compared it to di↵erent forecasting methods such as FNN and GRU. In addition, the results of the electricity price forecasting with LSTM are presented.

Furthermore, the smart grid simulation is carried in order demonstrate the gain of the proposed method and compared with the other existing methods. The results show a significant gain in terms of energy generation cost at distribution level and in terms of electricity consumption cost at the consumption level.

5 General conclusion

Load forecasting has gained increasing importance due to the advancement of smart grid technology. This project focuses on predicting hourly loads and electricity price one day in advance using Deep Learning techniques. Our approach introduces a flexible architecture that integrates various types of input features. These inputs are processed using LSTM model tailored to their specific characteristics. To evaluate our method, we analyze a dataset comprising several years of hourly load data. The experimental results demonstrate the e↵ectiveness of our approach.

Forecasting electricity price is also a crucial step in residential energy management. i.e., at consumption level, where the consumer should have a precise results to change his electricity consumption patterns to reduce his electricity consumption cost.

Forecasting load consumption and electricity price are a complex task due to their influence of numerous factors. Consequently, more training data is typically required to improve performance from a data standpoint. Additionally, incorporating additional relevant features such as hourly temperature and humidity can enhance the accuracy of load and electricity price forecasting.

Keeping up with the rapid development of Deep Learning methods is crucial, as novel models continue to emerge. One promising direction is the application of new mechanisms in deep learning such as federated learning, which have achieved considerable success in various domains. In future research, we plan to explore the potential of incorporating federated learning mechanisms into smart grid energy management to improve the overall performance.

References

[1] X.-F. Wang, Y. Mu, and R.-M. Chen, "An efficient privacy-preserving aggregation and billing protocol for smart grid," *Security and Communication Networks*, vol. 9, no. 17, pp. 4536–4547, 2016.

[2] I. Zabalza and J. Gómez-Aleixandre, "Smart grids: An overview," *Renewable and Sustainable Energy Reviews*, vol. 69, pp. 436–446, 2017.

[3] J.-H. Han, W.-J. Lee, S.-Y. Lee, and J.-K. Choi, "Electric vehicles in smart grid: a review of coordinated approaches for grid transactions," *Energies*, vol. 11, no. 5, p. 1125, 2018.

[4] F. Li, Y. Li, and Y. Fu, "Smart grid technologies: a comprehensive review," *Journal of Modern Power Systems and Clean Energy*, vol. 8, no. 1, pp. 1–17, 2020.

[5] H. Farhangi, "The path of the smart grid," *IEEE Power and Energy Magazine*, vol. 8, no. 1, pp. 18–28, 2010.

[6] V. K. Sood and S. Chakrabarti, *Smart grid: technology and applications*. John Wiley & Sons, 2017.

[7] C. Guo, J. Jiang, J. Zhang, and Y. Zhang, *Big Data Analytics for Smart Grids*. Springer, 2017.

[8] M. Maknouninejad and J. P. Catal ao, "Multi-objective optimal sizing of transmission lines in smart grids," *Energies*, vol. 11, no. 10, p. 2774, 2018.

[9] Y. Chen, J. Zhang, and Y. Zhang, "Big data analytics for dynamic energy management in smart grids," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 1743–1753, 2017.

[10] E. Vrettos and E. Kardakos, "Smart grid architecture in a nutshell," *Electric Power Systems Research*, vol. 151, pp. 294–302, 2017.

[11] P. Kundur, N. Balu, and M. G. Lauby, *Power system stability and control*. McGraw Hill Professional, 2016.

[12] C. Cecati, P. Siano, and F. Calogero, *Smart Grid: Technologies, Applications and Impact on the Power System*. Springer Science & Business Media, 2013.

[13] S. K. Rathor and D. Saxena, "Energy management system for smart grid: An overview and key issues," *International Journal of Energy Research*, vol. 44, no. 6, pp. 4067–4109, 2020.

[14] I. Naidji., O. Mosbahi., M. Khalgui., and A. Bachir., "Cooperative energy management software for networked microgrids," in *Proceedings of the 14th International Conference on Software Technologies - Volume 1: ICSOFT,*, pp. 428–438, INSTICC, SciTePress, 2019.

[15] M. Amer, A. Naaman, N. M'Sirdi, and A. El-Zonkoly, "Smart home energy management systems survey," in *International Conference on Renewable Energies for Developing Countries 2014*, pp. 167–173, IEEE, 2014.

[16] I. Naidji, M. B. Smida, M. Khalgui, and A. Bachir., "Multi agent system-based approach for enhancing cyber-physical security in smart grids," in *Proceedings of the the 33rd Annual European Simulation and Modelling Conference*, pp. 177–182, ETI, EUROSIS, 2019.

[17] K. T. Chui, M. D. Lytras, and A. Visvizi, "Energy sustainability in smart cities: Artificial intelligence, smart monitoring, and optimization of energy consumption," *Energies*, vol. 11, no. 11, 2018.

[18] H. Herath and M. Mittal, "Adoption of artificial intelligence in smart cities: A comprehensive review," *International Journal of Information Management Data Insights*, vol. 2, no. 1, p. 100076, 2022.

[19] P. K. Agarwal, J. Gurjar, A. K. Agarwal, and R. Birla, "Application of artificial intelligence for development of intelligent transport system in smart cities," *Journal of Tra c and Transportation Engineering*, vol. 1, no. 1, pp. 20–30, 2015.

[20] A. C. Şerban and M. D. Lytras, "Artificial intelligence for smart renewable energy sector in europe—smart energy infrastructures for next generation smart cities," *IEEE Access*, vol. 8, pp. 77364–77377, 2020.

[21] I. Naidji, M. Ben Smida, M. Khalgui, A. Bachir, Z. Li, and N. Wu, "Efficient allocation strategy of energy storage systems in power grids considering contingencies," *IEEE Access*, vol. 7, pp. 186378–186392, 2019.

[22] W. He, "Load forecasting via deep neural networks," *Procedia Computer Science*, vol. 122, pp. 308–314, 2017.

[23] I. Naidji, M. B. Smida, M. Khalgui, and A. Bachir, "Non cooperative game theoretic approach for residential energy management in smart grid," in *The 32nd Annual European Simulation and Modelling Conference*, (Ghent, Belgium), pp. 164–170, 2018.

[24] R. D. Zimmerman and C. E. Murillo-Sánchez, "Matpower 6.0 user's manual," *Power Systems Engineering Research Center*, vol. 9, 2016.

الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة محمد خيضر بسكرة

صورة العالمة التجارية

االسم التجاري

Smart DZ Power

السنة الجامعية

2022 _ 2023

عنوان المشروع:

Ai-based Energy management system for smart grid network

مشروع لنيل شهادة مؤسسة ناشئة في إطار القرار الوزاري 1275

بطاقة معلومات:

حول فريق الإشراف وفريق العمل

1- فريق الإشراف:

فريق الإشراف	
(01): المشرف الرئيسي نعيجي إلياس	**:** التخصص إعلام آلي

2- فريق العمل:

فريق المشروع	التخصص	الكلية
الطالب: سالطنية نعيمة	(RTIC) إعلام آلي	كلية العلوم الدقيقة وعلوم الطبيعة والحياة
الطالب: طوارف نور الهدى	(RTIC) إعلام آلي	كلية العلوم الدقيقة وعلوم الطبيعة والحياة

فهرس المحتويات

عنوان المشروع: Ai-based Energy management system for smart grid network

المحور الثاني: الجوانب االبتكارية

المحور الثالث: التحليل االستراتيجي للسوق

المحور الرابع: خطة اإلنتاج والتنظيم

المحور الخامس: الخطة المالية

المحور السادس: النموذج االولي التجريبي

المحور األول: تقديم المشروع

1. فكرة المشروع)الحل المقترح(

تتكون إدارة الطاقة في الشبكة الكهربائية الذكية من تقنيات وتكنولوجيات مختلفة تهدف إلى التوليد ، التوزيع واالستهالك األمثل للكهرباء. وهذا ينطوي على تكامل نظم متقدمة للرصد والتحكم واالتصاالت لتمكين عمليات الطاقة الفعالة والموثوقة.

في اآلونة األخيرة، يلعب النمو السريع لتكنولوجيا المعلومات واالتصاالت والبنية التحتية المتقدمة للقياس)AMI(، والتي تشمل العدادات الذكية وشبكات االتصاالت دو را رئيسيًا في تطوير الشبكة الذكية.

عنوان المشروع: Ai-based Energy management system for smart grid network

لهذا السبب، يُظهر الذكاء الاصطناعي إمكانات كبيرة لتحسين إدارة الطاقة في الشبكة الذكية والتي يمكن أن تحسنها بمستوياتها المختلفة من التوليد إلى الاستهلاك.

يهدف هذا المشروع إلى توفير حل قائم على الذكاء الاصطناعي للشبكة الذكية على مستوى التوزيع والاستهلاك. وينبغي أن يكفل الحل نتيجة موثوقة لتحسين عملية إدارة الطاقة التي تستند إلى بيانات احتمالية.

يجب أن يضمن الحل على مستوى التوزيع التنبؤ بالحمل لإدارة الطاقة، وهي مشكلة تنبؤ متسلسلة زمنية معقدة، تعتمد على ميزات تقنية مختلفة مثل درجة الحرارة والوقت من اليوم... وما إلى ذلك.

وينبغي أن يكفل الحل على مستوى الاستهلاك التنبؤ بأسعار الكهرباء لإدارة الطاقة على مستوى الاستهلاك، وهي أيضا مشكلة متسلسلة زمنية معقدة، تعتمد على سمات مختلفة؛ بحيث تصل تكلفة استهلاك الكهرباء إلى الحد الأدنى. سيتم تثبيت هذا الحل على مستوى عداد ذكي موصول بحساسات، يقوم برصد الطاقة المستهلكة من طرف جميع الأجهزة الكهرو منزلية وكذا التحكم فيها بواسطة تطبيق إلكتروني.

بالنسبة لكل حل، يجب تطوير نموذج تنبؤ متقدم لتحقيق نتائج أفضل من حيث تكلفة الطاقة.

2. القيم المقترحة

- تنظيم الطاقة الكهربائية بصفة ذكية.
- التقليل من ضياع الطاقة الكهربائية.
- التوجه نحو استغلال الطاقات المتجددة.
- التحكم في استهلاك الطاقة الكهربائية.
- التنبؤ بكمية الطاقة المراد استهلاكها.
- انخفاض تكلفة استهلاك الطاقة الكهربائية.

3. فريق العمل

يتكون فريق المشروع من الآتي:

- الأستاذ المشرف **نعيجي إلياس**، تخصص **إعلام آلي**
- الطالبة **سالطنية نعيمة**، تخصص **شبكات وتكنولوجيا الإعلام والاتصال –ماستر 2 إعلام آلي–**
- الطالبة **طوارف نور الهدى**، تخصص **شبكات وتكنولوجيا الإعلام والاتصال –ماستر 2 إعلام آلي–**

4. أهداف المشروع:

- إيجاد حلول للتقليل من ضياع الطاقة.

عنوان المشروع: Ai-based Energy management system for smart grid network

- التقليل من تكلفة استهلاك الكهرباء.
- التنبؤ بكمية الكهرباء المراد توزيعها.
- التنبؤ بتكلفة الكهرباء المستهلكة.

5. جدول زمني لتحقيق للمشروع:

	الأعمال	الأشهر									
		1	2	3	4	5	6	7	8	9	10
1	الدراسات الأولية	(✓)	(✓)								
2	البحث عن قواعد البيانات الخاصة بالكهرباء		(✓)	(✓)							
3	تحليل المعطيات				(✓)	(✓)					
4	التأكد من صحة التنبؤات					(✓)	(✓)				
5	طلب التجهيزات من الخارج							(✓)			
6	تركيب المعدات								(✓)		
7	تصميم واجهة التطبيق								(✓)		
8	تثبيت النتائج والتحليلات على الأجهزة									(✓)	
9	بداية انتاج أول منتج									(✓)	
10	بداية تحميل أول تطبيق										(✓)

المحور الثاني: الجوانب الابتكارية

قد تكون بعض القيم المقترحة مبتكرة بحيث:

✓ **استخدام التكنولوجيا الحديثة:**

- يعتمد المشروع على استخدام التكنولوجيا الحديثة كالذكاء الاصطناعي، وتقنيات الوقت الحقيقي للأحداث والمعلومات الخاصة بالطاقة الكهربائية ومستخدميها.

✓ قدرة المستهلك على التحكم في الطاقة من خلال التحكم في الأجهزة الكهرو منزلية عن بعد.

✓ ارسال اشعارات في حالة ارتفاع استهلاك الكهرباء في جهاز معين.

✓ التنبيه عن وجود خلل كهربائي أو تسرب للغاز.

✓ تقنيات التعلم العميق: يمكن استخدام تقنيات التعلم العميق مثل الشبكات العصبية العميقة لتحليل البيانات الكبيرة المتعلقة بالطاقة الكهربائية. يمكن أن تساعد هذه التقنيات في تحسين دقة التوقعات.

✓ التحليل الشامل للمعطيات: يمكن تنفيذ تحليل شامل لمجموعة واسعة من البيانات ذات الصلة بتوزيع واستهلاك الكهرباء و كذلك المتعلقة بسعر الطاقة الكهربائية المستهلكة.

المحور الثالث: التحليل الاستراتيجي للسوق

عنوان المشروع: Ai-based Energy management system for smart grid network

يتضمن تحليل المتغيرات الكلية (**PESTEL**) وتحليل القوى التنافسية (**POTER**) وتحليل (**SWOT**):

أولا-تحليل المتغيرات الكلية(PESTEL) : هي أداة تستخدم من طرف المؤسسات لتحليل ومراقبة العوامل البيئية التسويقية الخارجية التي تؤثر على هذه المؤسسات، ويتكون من العناصر التالية:

العوامل السياسية - المنافسة -التجارة الالكترونية -وضع إجراءات للتحفيز والتشجيع بالنسبة للعمليات والمشاريع التي تساهم في تحسين النجاعة الطاقوية وترقية الطاقات المتجددة. - تم إنشاء الصندوق الوطني للتحكم في الطاقة ليساهم في تمويل المشاريع.	العوامل الاقتصادية -القوة الشرائية للمستهلكين. - تمتلك الجزائر إمكانيات ضخمة في مختلف مصادر الطاقة المتجددة، خصوصا الطاقة الشمسية. - التسهيلات التي منحتها الدولة للمؤسسات الناشئة. التكلفة المتوقعة لتشغيل وصيانة البنية التحتية اللازمة لتطبيق الشبكة الذكية وإدارة الطاقة. -انخفاض تكلفة استهلاك الكهرباء.	العوامل الاجتماعية - المواقف اتجاه الطاقات المتجددة - مراعات الطبقات الاجتماعية من حيث تكلفة الطاقة المستهلكة. - مدى قبول المستهلكين والمجتمع للتكنولوجيا الذكية وتوفير الحلول الذكية لإدارة الطاقة.
العوامل التكنولوجية -الأنترنت. -الذكاء الاصطناعي.- التقدم في تكنولوجيا الذكاء الاصطناعي والتحليل البياني والتعلم الآلي لتوفير تنبؤات دقيقة لإدارة الطاقة. -امكانية وصول المستهلك إلى التكنولوجيا. - توفر أجهزة الاستشعار والعدادات الذكية التي تجمع البيانات الدقيقة حول استهلاك الطاقة والأحمال الكهربائية.	العوامل البيئية - المواقف اتجاه الطاقة المتجددة ودعمها -تأثير استخدام تقنيات إدارة الطاقة الذكية على الاستدامة البيئية وتقليل الانبعاثات الكربونية - توفير الحلول الذكية لإدارة الطاقة التي تساهم في تحسين كفاءة استخدام الطاقة وتقليل الهدر.	العوامل القانونية -حماية حقوق الملكية الفكرية والبراءات والعلامات التجارية المتعلقة بتقنيات الذكاء الصناعي والشبكات الذكية. - قانون الضرائب الملغاة للسنوات الأولى الخاص بالمؤسسات الناشئة.

ثانيا-تحليل القوى التنافسية (PORTER): إطار عمل بسيط لتقييم وضع أي مؤسسة والقوى التنافسية الخاصة بها، ويقوم هذا الإطار على فكرة أن هناك خمس قوى تنافسية أساسية تشكل كل صناعة، وتساعد على تحديد حدة المنافسة وجاذبية السوق، مما يساعد على فهم مدى قوة المركز التنافسي الحالي للمؤسسة، ومدى قوة المركز الذي تستطيع الوصول إليها، ويتكون من العناصر التالية:

1- تهديد المنافسة الجديدة:

- تهديدات دخول شركات جديدة لسوق الشبكة الذكية وحلول الذكاء االصطناعي الخاصة بالطاقة الكهربائية في الجزائر.
- تطوير المؤسسة لميزات تريدة وتكنولوجيا متقدمة للحفاظ على تنافسيتها في السوق.
- العمل على الوصول إلى ألعلى درجات الدقة في المعطيات للحصول على أفضل النتائج.

2- قوة العمالء:

- قوة المناوضة على األسعار لدى المشترين.
- مناوضة وتحديد الشروط من قبل العمالء أصحاب المجال)الشركات الكهربائية أو نقاط بيع المعدات الكهربائية(لخبرتهم التامة في مجال الطاقة الكهربائية.
- تقديم المؤسسة قيمة مضافة وحل تريد يلبي احتياجات ومتطلبات العمالء كالعمل على الوصول إلى االستهالك األمثل للكهرباء وبتكلفة أقل.
- التواصل المباشر مع العمالء من خالل البيانات الشخصية أو اللقاء معهم بصفة شخصية.

3- قوة الموردين:

- تنويع المؤسسة لمصادرها والتناوض بشأن أسعار المواد األولية وشروط التوريد للحصول على أفضل عروض.
- المؤسسة ستحتاج إلى التعاون مع موردين لتوفير التقنيات والمكونات االلزمة للحل الذكي.
- يمكن للمؤسسة التناوض مع الموردين بشأن األسعار وشروط التوريد لضمان توفر الموارد بتكلفة معقولة وجودة عالية.

4- تهديد المنتجات البديلة:

- وجود مؤسسات أخرى في الجزائر تهتم بتنظيم الطاقة والعدادات الذكية.
- عدم تأقلم بعض الفئات مع التطور التكنولوجي واعتمادها على الطرق التقليدية.
- يجب أن يتميز الحل الذكي بقوة الذكاء االصطناعي وتحليل البيانات االحتمالية لتحسين أداء إدارة الطاقة وتوفير نتائج موثوقة.

5- تهديد دخول منافسين جدد:

- تسهيل ودعم الدولة الجزائرية للمؤسسات الناشئة وخاصة في مجال التكنولوجيا.
- حجم المنافسين في السوق ال يعتبر بالكبير.
- عمل المؤسسة على تطوير استراتيجية تنافسية تريدة ومزايا تميزها عن المنافسين للفوز بحصة السوق والعمالء.

ثالثا-تحليل (SWOT): وهي طريقة للتحليل االستراتيجي تساعد على تحليل البيئة الداخلية للمؤسسة والمتمثلة بنقاط قوتها ونقاط ضعفها من جانب، وما يقابلها من تحليل للبيئة الخارجية المتمثلة بالفرص المتاحة والتهديدات التي قد تواجهها، وقد سميت هذه المصفوفة بتحليلSWOT اختصاراً للحرف األول من كل عنصر من عناصرها:

عنوان المشروع: Ai-based Energy management system for smart grid network

نقاط الضعف (Weakness)	نقاط القوة (Strengths)
- التحديات المتعلقة بتوفير بنية تحتية مالئمة وتثبيت العدادات الذكية والمستشعرات المتصلة في المنازل والمباني. - قلة االهتمام باستخدام المصادر المتجددة إلنتاج الطاقة والفهم الخاطئ لطبيعة عمل وتطبيقات تكنولوجيا الطاقة المتجددة من قبل األطراف المعنية والمجتمع ككل. - ضعف الهياكل التصنيعية األساسية ونقص القدرات الفنية والتقنية االلزمة لتطبيق تكنولوجيا الطاقة المتجددة. -عدم تقبل الجهاز من طرف شركات الكهرباء. -إمكانية عدم توفر المواد األولية للتصنيع (الحساسات...). -التمويل.	-توجه الدولة نحو الرقمنة والتطور التكنولوجي. -القدرة على تحليل واستخدام البيانات الكبيرة وتقنيات الذكاء االصطناعي لتوفير تنبؤ دقيق بالحمل وتحسين إدارة الطاقة. - قدرة المشروع على تحسين كفاءة استخدام الطاقة وتقليل الهدر. - القدرة على توفير حل موثوق ودقيق للتنبؤ بأسعار الكهرباء وتحسين استهالك الطاقة. - تم إنشاء الصندوق الوطني للتحكم ني الطاقة ليساهم في تمويل المشاريع. -إمكانية إنتاج الجهاز.
التهديدات (Threats)	الفرص (Opportunities)
- المنافسة القوية من شركات أخرى تعمل ني مجال حلول الذكاء االصطناعي وإدارة الطاقة. - التحديات التمويلية واالستثمارية في تطوير وتنفيذ مشروع توفير حل ذكاء اصطناعي للشبكة الذكية. -ارتفاع سعر المواد األولية مما يؤدي إلى ارتفاع سعر الجهاز.	-الدعم والتسهيالت المتاحة ألصحاب المؤسسات الناشئة من طرف الدولة. - التعاون المحتمل مع شركات الطاقة والمزودين لتطبيق الحلول الذكية. - تطور قدرات تحليل البيانات والتعلم اآللي التي تساهم في تحسين دقة التنبؤ بالحمل وأسعار الكهرباء. -تشجيع الدولة على التوجه نحو الطاقات المتجددة وخاصة الشمسية.

رابعا: المزيج التسويقي

- هو عبارة عن مجموعة من الخطط والسياسات والعمليات التي تمارسها اإلدارة التسويقية بهدف إشباع حاجات المستهلكين، وتحقيق الربح للمؤسسة، ويتضمن المزيج التسويقي أربع عناصر في المؤسسة الناجية متمثلة في المنتج والسعر والترويج والتوزيع، وهذه المجموعات تتأثر ببعضها البعض

- عناصر المزيج التسويقي للمؤسسات الخدمية.

1- المنتج (Product):

عنوان المشروع: Ai-based Energy management system for smart grid network

- يتعلق بتطوير حل قائم على الذكاء االصطناعي للشبكة الذكية إلدارة الطاقة وتحسين استهالك الكهرباء.
- الوصول إلى حل يكون الحل قاد ًرا على التنبؤ بالحمل وأسعار الكهرباء بشكل موثوق ودقيق. - يشمل صناعة وتثبيت عداد ذكي موصول بحساسات لرصد استهالك الطاقة والتحكم فيه بواسطة تطبيق إلكتروني.

2- السعر (Prix):

- السعر المحدد مناسب للحل الذكي المقدم، مع مراعاة قيمته وفوائده المتوقعة. - يحدد ارتفاع أو انخفاض السعر حسب تكلفة المواد األولية)عالقة طردية(، الطلب على المنتج)عالقة عكسية(أو المؤسسات المنافسة.

3- التوزيع)Place(:

- **مباشر:**
- يتم في غالب األحيان عند التعامل مع أصحاب الشركات. -
- تزويد العميل مباشرة بالخدمة أو المنتوج.
- **غير مباشر:**
- يتم االستعانة بوسيط)تجار التجزئة أو الجملة(للتواصل مع العميل في حالة ما إذا كان مواطن عادي.
-

4- الترويج)Promotion(:

- تنظيم لقاءات مع أصحاب الشركات الكهربائية إلقناعهم بالمشروع.
- االعتماد على اعالنات ممولة على منصات مواقع التواصل االجتماعي،
- استخدام وسائل اإلعالم المختلفة كالتلفاز والراديو للوصول إلى الجمهور المستهدف وتعزيز الوعي به.
- انشاء صفحات على مواقع التواصل االجتماعي خاصة بالمؤسسة.
- القيام بحمالت ترويجية للتوعية بفوائد الحل الذكي وكيفية تحسين إدارة الطاقة وتوفير الكهرباء.

خامسا: االستراتيجيات التسويقية.

تتمثل االستراتيجيات التسويقية في ثالث استراتيجيات كما يلي:

- استراتيجية القيادة بالتكلفة:

- هي استراتيجية نعتمدها في مشروعنا لتحقيق تفوق تنافسي من خالل تقديم المنتجات أو الخدمات بأقل تكلفة ممكنة. نحن نركز على تحسين الكفاءة في اإلنتاج وتحسين العمليات الداخلية لتقليل التكاليف وتحقيق أفضل قيمة للعمالء.
- لتحقيق هذه االستراتيجية، نعمل على تحليل وتحسين عمليات المشروع بحيث تكون أكثر كفاءة وتوفيرا للتكاليف. نقوم بتقديم التدريب والتعليم المستمر لفريق العمل لزيادة مهاراتهم وتحسين أدائهم في سبيل تقليل التكاليف العملية وزيادة اإلنتاجية.
- نستثمر أيضا في تكنولوجيا اإلنتاج المتقدمة والمبتكرة لتحسين عمليات اإلنتاج وتقليل التكاليف. يمكن أن تشمل هذه التكنولوجيا استخدام اآلالت والمعدات المتطورة وتنفيذ العمليات التي تزيد من كفاءة اإلنتاج وتقلل من الخطأ البشري .بشكل عام، نسعى جاهدين لتقديم المنتجات بجودة عالية وبأسعار تنافسية في السوق. أعتمد على تحليل البيانات واستخالص األرقام لتحديد تكاليف اإلنتاج وضبطها بشكل مناسب، وذلك لتحقيق ربحية مستدامة وتلبية احتياجات العمالء بأفضل قيمة من خالل تحليل البيانات المالية واالقتصادية وفهم احتياجات السوق، نحن نعمل على تحديد سعر منتجاتنا بدقة لضمان تحقيق توازن بين القيمة المقدمة والتكلفة. نحن نسعى لتحقيق ميزة تكلفة على المنافسين من خالل التحكم في التكاليف وتقليل المصاريف غير الضرورية .بالنسبة للتمويل، نقوم بإعداد خطة مالية متكاملة تشمل تقديرات التكاليف المتوقعة لإلنتاج والتسويق والتوزيع. أيضا،

نقوم بإدارة التدفقات النقدية بعناية لضمان توفر السيولة الكافية لتلبية االحتياجات اليومية للمشروع.

- استراتيجية التميز:

- ، نعتمد على تقديم منتجات وخدمات فريدة ومتميزة عن المنافسين في السوق. ونسعى إلى انشاء عالمة تجارية قوية تتميز بالجودة واالبتكار والتصميم المتفرد.
- نحرص على تحليل وفهم احتياجات ورغبات العمالء بدقة. بنا ء على هذا التحليل، ونقوم بتطوير منتجات وخدمات فريدة تلبي تلك االحتياجات بطرق مبتكرة ومتنوعة.
- نقوم باالهتمام بجودة المنتجات والخدمات التي نقدمها والعمل على تحسين العمليات والمعايير الداخلية لضمان جودة عالية.
- نعمل على تطوير عوامل التميز التي تميز منتجاتي عن المنافسين. قد يكون ذلك عبر توفير مزايا فنية أو تقنية متقدمة، أو تصميم فريد وجذاب، أو تجربة عمالء متميزة، أو خدمة ما بعد البيع المميزة.

- استراتيجية التركيز:

- عند تبني استراتيجية التركيز، نقوم بتحديد الشريحة المستهدفة بدقة وفهم احتياجاتها ومتطلباتها الخاصة.
- نعمل على تصميم منتجات وخدمات تلبي تلك االحتياجات بشكل مميز وفريد.

- يمكن أن تكون استراتيجية التركيز موجهة نحو الأسواق الجغرافية مثل استهداف سوق محلي معين أو سوق إقليمي، أو استهداف مؤسسات وشركات معينة تهتم بالطاقات المستهلكة وخاصة الشركات الصناعية.
- استراتيجية التركيز تمنحنا القدرة على تخصيص الموارد والجهود بشكل فعال نحو الشريحة المستهدفة، مما يساعدنا على تحقيق تفوق تنافسي في هذا القطاع المحدد. كما أنها تمكننا من تقديم منتجات وخدمات تفوق توقعات العملاء في هذه الشريحة المستهدفة.

المحور الرابع: خطة الإنتاج والتنظيم

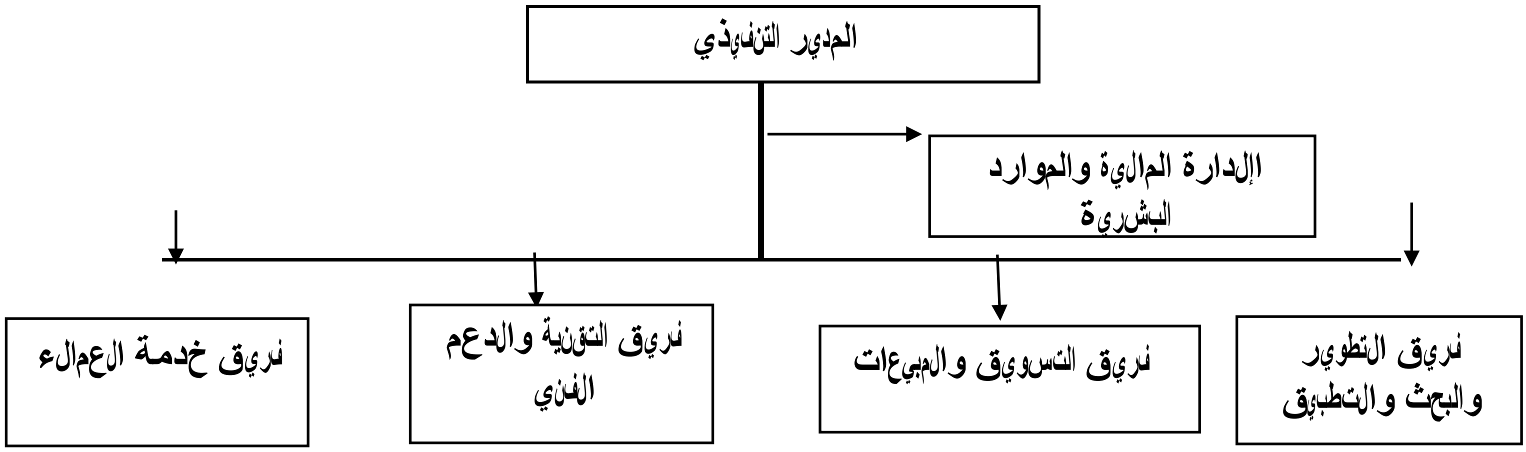
أولا-خطة الإنتاج:

يمثل المخطط الإنتاجي الأخذ بعين الاعتبار الطلب المستقبلي على سلعة معينة، وعلى هذا الأساس يتم تعبئة الطاقات الإنتاجية اللازمة لمواجهة الطلب، إن هذا النشاط يتطلب التوقع بالمبيعات المستقبلية، وترجمة هذا التوقع في مختلف المتطلبات الإنتاجية والعمل للحصول على هذه المتطلبات، وتتمثل خطوات اعداد المخطط الإنتاجي فيما يلي:

- تحديد المتطلبات والمواصفات: تحديد المتطلبات الفنية والوظيفية للحل القائم على الذكاء الاصطناعي للشبكة الكهربائية الذكية على مستوى التوزيع والاستهلاك وتحديد الميزات والوظائف المحددة التي يجب توفيرها لتحسين عملية إدارة الطاقة.
- تصميم الحل: تصميم الهيكل العام والتفاصيل التقنية للحل، بما في ذلك الخوارزميات والنماذج المستخدمة في التنبؤ بالحمل وأسعار الكهرباء وتحديد الاعتمادات التقنية والموارد اللازمة لتطبيق الحل.
- تطوير البرمجيات والأجهزة: تطوير وبرمجة البرمجيات المطلوبة للحل واختبارها وتحسينها وتكوين الأجهزة الضرورية، بما في ذلك عدادات الكهرباء الذكية والحساسات المرتبطة بها.
- التجميع والتثبيت: تجميع الأجهزة وتثبيتها على مستوى العدادات الذكية في المواقع المناسبة وتكوين الاتصالات والشبكات اللازمة لنقل البيانات بين العدادات والتطبيق الإلكتروني.
- تدريب ودعم العملاء: تقديم تدريب شامل للعملاء حول استخدام الحل والتطبيق الإلكتروني وتقديم الدعم الفني والمساعدة المستمرة للعملاء للتأكد من استخدامهم الفعال للحل وتحقيق الفوائد المرجوة.

عنوان المشروع: Ai-based Energy management system for smart grid network

ثانيا-المخطط التنظيمي:



المحور الخامس: الخطة المالية (PLAN FINANCIER)

المخطط المالي يترجم التكاليف في شكل كمي ونقدي

أولا-تكاليف المشروع واهتلاك الاستثمار.

تكاليف المشروع: تتمثل التكاليف الإجمالية للمشروع في التكاليف الاستثمارية والتكاليف التشغيلية:

التكاليف الاستثمارية:

الأصول	التكلفة
المباني	/
الآلات والمعدات	**100000DA**
الأثاث	/
رأس المال العامل	**10000DA**
المجموع	**110000DA**

التكاليف التشغيلية:

الأصول	التكلفة
مواد أولية	**10540DA**
أجور	/
الهاتف والانترنت	**36000DA**
الكهرباء والماء	/
المجموع	**46540DA**

1- الهيكل التمويلي: يتم تمويل المشروع بعدة طرق إما بالاعتماد الكلي على الأموال الخاصة لصاحب المشروع وهذا ما يسمى بالتمويل الذاتي، أو الاستعانة بأحد المؤسسات المالية وذلك

عنوان المشروع: Ai-based Energy management system for smart grid network

عن طريق التمويل الثنائي او الثالثي، أما بالنسبة لمشروعنا (سيكون بالتمويل).

	PREVISION					
Produit A destiné Client	**N**	**N+1**	**N+2**	**N+3**	**N+4**	**N+5**
Quantité produit A	**10**	**25**	**50**	**75**	**100**	**250**
Prix HT produit A	**25000DA**					
Ventes produit A	**3**	**16**	**42**	**60**	**-93**	**220**
CHIFFRE D'AFFAIRES GLOBAL	**75000DA**	**400000DA**	**1050000DA**	**1500000DA**	**2325000DA**	**5500000DA**

الذاتي كما هو موضح في الجدول التالي:

البيان	النسبة	القيمة
أموال خاصة	**100%**	**110000DA**
القروض	**0%**	**0**
المجموع	**100%**	**110000DA**

المحور السادس: النموذج الاولي التجريبي

Ai-based Energy management system for smart grid network :عنوان المشروع

الملحق رقم 04: نموذج العمل التجاري

<table>
<tr>
<td rowspan="2">الشراكات الرئيسية
Key Partners
ركة الكهرباء
-تجار الجملة</td>
<td>الأنشطة الرئيسية
Key Activities
-شراء حساسات
-شراء معدات خاصة بالجهاز.
-البرمجة
-تركيب الأجهزة
-البيع
-الصيانة</td>
<td rowspan="2">القيم المقترحة
Value Proposition
-تكلفة منخفضة للطاقة

الكهربائية. -التنبيه عند اجتياز
التكلفة قيمة

معينة. -التنبيه عند اشتغال
الأجهزة مدة
طويلة.
-التنبيه بتسرب الغاز.</td>
<td>العلاقات مع العملاء
Customer Relationships
-خدمة العميل
-توفير خدمة الصيانة</td>
<td rowspan="2">شرائح العملاء
Customer Segments
- المواطن العادي
- أصحاب شركات الطاقة

الكهربائية - أصحاب المؤسسات
الصناعية</td>
</tr>
<tr>
<td>الموارد الرئيسية
Key Ressources
مال كهرباء
-حساسات -
الحاسوب
-أجهزة أخرى)Arduino،
) relay،câbles</td>
<td>القنوات
Channels
-توزيع غير مباشر عن طريق تجار
التجزئة أو الجملة.
-توزيع مباشر لأصحاب شركات الطاقة
والشركات الصناعية.</td>
</tr>
<tr>
<td colspan="3">هيكل التكاليف
Cost Structure
-تكاليف الكهرباء
-تكاليف إيجار المكان
-التجهيزات المكتبية -
أجور الموظفين
-الضرائب
-تكاليف الحساسات، arduino، relay...</td>
<td colspan="2">مصادر الإيرادات
Revenue Streams

يع الجهاز
-الصيانة
-تنظيم الطاقة لشركات الكهرباء</td>
</tr>
</table>