The University of South Bohemia in České Budějovice
Faculty of Science

Smart Grid Energy Forecasting: Enhancing Forecast Performance through Federated and Split Learning

Master´s thesis

Marwan Ahmed

Advisor: Prof. Dr. Andreas Kassler

České Budějovice 2024

M.M.Ahmed, “Smart Grid Energy Forecasting: Enhancing Forecast Performance through Federated and Split Learning” MS. thesis in English, Faculty of Science, University of South Bohemia, Ceské Budéjvice, Czech Republic and Faculty of Applied Computer Science, Deggendorf Institute of Technology, Deggendorf, Germany, Jan. 2024, p.71

Annotation:
Federated and split learning models were applied to forecast power consumption in smart grids, with a focus on integrating renewable energy sources while prioritizing data privacy, computational efficiency, and accuracy. The study conducted a comparative evaluation of these two methods, investigating various parameters influencing the performance of split learning**.**

I declare that I am the author of this qualification thesis and that in writing it I have used the sources and literature displayed in the list of used sources only.

České Budějovice	31. January 2024	Marwan mustafa
	Date	Signature

Abstract

This thesis delves into the deployment of federated and split learning models for forecasting power consumption within smart grids, with a focus on integrating renewable energy sources and ensuring data privacy. Accurate predictions of power consumption are crucial for managing energy exchanges effectively, especially with the variability introduced by renewables. This need becomes even more critical when dealing with sensitive data from Prosumers equipped with photovoltaic (PV) systems, highlighting the importance of maintaining privacy.

Our research applies both split learning and federated learning models to predict Prosumer power consumption, aiming to enhance smart grid operations' efficiency and privacy. The study addresses the challenge of forecasting power usage accurately, managing the computational demands of advanced learning techniques, and handling sensitive data from diverse Prosumers. A pivotal aspect of this research is the implementation of a split learning model, focusing on the strategic division of the neural network layer between client and server, which significantly impacts the model's performance, computational requirements, and privacy protections.

Through a comparative analysis between federated and split learning experiments, split learning has proven to be the more effective method for forecasting power consumption in smart grids. Despite the privacy and decentralization benefits of federated learning, it struggled to accurately capture the varied energy usage patterns across different Prosumers, resulting in a mean absolute error (MAE) of 87 watts. This contrasted with the initial split learning experiment, which established a strong baseline with a significantly lower mean MAE of 8.94 watts, demonstrating its superior capability in balancing accuracy, computational efficiency, and privacy.

The subsequent split learning experiments, which varied the client-side complexity, further underscored split learning's advantages. Despite slight increases in MAE, split learning consistently outperformed the federated model, affirming its suitability for smart grid energy forecasting. The decision to focus on split learning was driven by its demonstrated flexibility in task distribution and its ability to optimize model performance effectively.

This thesis concludes that split learning, with its adaptability, efficiency, and lower error rate, stands as the optimal approach for forecasting power consumption in smart grids. It navigates the complexities of renewable energy integration, offering a promising solution for enhancing the sustainability and reliability of energy systems.

Acknowledgement

I extend my heartfelt gratitude to Prof. Dr. Andreas Kassler for his unwavering support and expert guidance throughout my master's thesis. His invaluable insights and encouragement have been fundamental to my research and academic growth.

I also wish to express my appreciation to Phil Aupke, Ph.D. candidate, for his assistance and contributions to my work. His expertise and collaborative spirit have greatly enriched my thesis.

My thanks go out to all the professors involved in my master's program for their role in this incredible journey. Their knowledge and passion have been instrumental in shaping my academic path.

Lastly, I am profoundly grateful to my family for their endless love, support, and belief in me. Their encouragement has been my constant source of strength and motivation.

Table of Contents

Chapter 1: Introduction

The global shift towards renewable energy sources is a critical component of contemporary energy policies, driven by an urgent need to foster sustainability and mitigate environmental impacts. This transition is underscored by international commitments, such as the Paris Agreement (UNFCCC, 2015), which aims to significantly reduce greenhouse gas emissions and combat climate change. Among renewable technologies, wind and solar power are particularly notable for their potential to replace traditional fossil fuel-based energy generation, thanks to ongoing technological advancements that have improved efficiency, reduced costs, and enhanced storage solutions. These developments not only contribute to the environmental benefits of renewable energy but also spur economic growth through job creation in the renewable sector and promote social equity by improving energy access. A vital part of seamlessly combining renewable energy sources into the power grid is accurate forecasting of production and consumption. Such forecasts designate grid operators to manage oscillations in energy supply and demand effectively, optimizing grid operations, reducing imbalances, and facilitating strategic decisions regarding the necessity for backup power solutions. This ability is especially crucial as the energy sector moves towards a more distributed and renewable-centric model, where the ability to predict power flows becomes integral to system reliability and efficiency.

Ensuring data privacy is now just as crucial as accurate power forecasting. With the widespread adoption of smart meters and sophisticated monitoring technologies, an extensive amount of data is being gathered, exposing detailed insights into personal energy consumption behaviors. Safeguarding this sensitive data against unauthorized access becomes a top priority, as any compromise could result in privacy violations and significant security risks for individuals. This intersection of data utility and privacy protection underscores a growing concern in the energy sector, highlighting the need for solutions that can balance effective energy management with the confidentiality of consumer information.

Innovative methodologies are essential to achieve reliable power forecasting while preserving data privacy. These methods must harness the wealth of data generated by smart grid technologies without exposing private information. One such innovative approach is the exploration of split learning and federated learning models. These models represent a paradigm shift in machine learning, offering a decentralized framework that allows for collaborative model training across various data sources without necessitating the centralization of sensitive data. By keeping data localized and preventing its direct sharing, these models not only enhance the predictive capabilities of power forecasting systems but also introduce a robust mechanism for protecting user privacy.

1.1 Background

Navigating the evolving landscape of grid management and energy flow prediction, particularly with the increasing integration of renewable energy sources like solar power and wind, highlights the limitations of traditional machine learning (ML) techniques. Historically, methods such as support vector machines, decision trees, and neural networks have been foundational in forecasting energy needs by leveraging vast historical data and weather patterns. Alongside these, modern machine learning methods like Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs) are gaining traction. LSTMs, with their ability to capture temporal dependencies, and CNNs, known for pattern recognition, offer advanced solutions to address the challenges posed by the variability and unpredictability of renewable energy sources. However, despite the advancements brought by LSTMs and CNNs, the efficiency of these methods is now being challenged by the evolving demands of energy systems. These conventional models, once tailored for stable and predictable power sources, now face hurdles with the unpredictability and variability of renewables. They struggle with scalability due to the sheer growth in distributed energy resources (DERs) and the volume of data these resources generate. Additionally, the heavy reliance on past data without adequately accounting for renewables' fluctuating nature may render these models less adaptable to new scenarios

This reliance on historical data and centralized data processing not only limits the adaptability and scalability of these forecasting models but also raises significant data privacy and security concerns. Traditional energy management systems often consolidate extensive data from diverse sources, such as consumer usage patterns and environmental conditions, centralizing it for processing. This centralization not only introduces bottlenecks in data analysis but also increases the risk of cyber threats and privacy breaches. It highlights the critical need for a shift towards more decentralized and privacy-conscious approaches in energy forecasting and grid management.

In response to these dual challenges, the exploration of split learning and federated learning represents a significant paradigm shift. These innovative approaches offer a decentralized framework that allows for collaborative model training across various data sources without necessitating the centralization of sensitive data. Keeping data localized and preventing its direct sharing, split learning, and federated learning not only enhances the predictive capabilities of power forecasting systems but also introduces a robust mechanism for protecting user privacy. This shift towards leveraging split learning and federated learning models addresses the pressing challenges of scalability, adaptability, and privacy in the context of renewable energy forecasting, marking a promising advancement in the field of energy management.

1.2 Research Problem and Motivation

In this thesis, the main challenge is to forecast power consumption in smart grids while emphasizing data privacy and computational efficiency. The task of accurately predicting power usage, especially when handling sensitive information from diverse Prosumers, poses a considerable challenge. This is made by the necessity to navigate the significant computational demands of advanced learning techniques required to analyze extensive and intricate datasets. Moreover, the research highlights the critical need for precise power consumption forecasts to ensure effective energy management and grid stability. The unpredictability of power usage patterns demands a forecasting solution that not only meets these challenges but also aids smart grid management in making well-informed decisions. Furthermore, this thesis seeks to explore the optimal approach for designing a forecasting framework that can successfully predict power consumption and provide a prediction range. This involves determining the most effective technique to accurately forecast power usage, thereby facilitating strategic planning and decision-making for smart grid operations. The motivation behind this thesis is driven by the critical need for a transition towards more sustainable energy systems, essential for combating climate change and ensuring energy security. With the increasing integration of unpredictable renewable energy sources like solar power, this thesis seeks innovative solutions—particularly federated and split learning models—that not only enhance energy consumption prediction accuracy but also protect individual data privacy. By addressing these challenges, the research contributes to the advancement of smart grids, facilitating a more effective integration of renewable energy sources and paving the way for a reliable and sustainable energy future.

1.3 Research Questions

This section outlines key research questions targeting the evaluation of split and federated learning models in smart grids. These questions aim to explore their effectiveness in energy forecasting, critical design elements, and comparative advantages. The goal is to uncover insights that improve energy management and address the challenges of renewable energy integration, data privacy, and computational efficiency in smart grid technologies.

Effectiveness of Split Learning for Energy Prediction:

- How effectively does split learning predict Prosumer power consumption within smart grids, and in what ways does it accommodate the dynamic nature of energy usage patterns?

Designing Split Learning Systems:

- Which parameters are crucial in enhancing the accuracy and efficiency of split learning models for energy consumption forecasting, and what strategies can be employed to optimize these parameters?

Comparative Analysis of Split Learning and Federated Learning:

- How does split learning compare to federated learning in terms of model performance, data privacy, and computational efficiency within the smart grid context?

1.4 Objective

This thesis aims to address the research question by considering the elements discussed earlier. To achieve this, the study is organized into the following key tasks:

- Implement Federated Learning Architecture: Develop a comprehensive federated learning framework tailored for energy consumption forecasting within smart grids. This involves designing the architecture to facilitate decentralized learning without compromising data privacy.
- Implement Split Learning Architecture with Varied Split Layer Positions: Construct a split learning model for energy forecasting, experimenting with different positions for the split layer. This task aims to identify how the layer's position affects the model's performance, computational demand, and privacy implications.
- Analyze and Compare Results of Each Architecture: Conduct a thorough analysis of the performance of both the federated and split learning architectures. This comparison will focus on evaluating their predictive accuracy, efficiency, and ability to maintain data privacy. The analysis will also explore the scalability of the models and their suitability for real-world smart grid applications.

Through these objectives, the thesis intends to provide a detailed evaluation of federated and split learning models, offering insights into their applicability and effectiveness in smart grid energy forecasting.

1.5 Methodology

1.5.1 Overview

This study evaluates split and federated learning models for smart grid power consumption forecasting, focusing on accuracy, efficiency, and privacy. The methodology involves setting up experiments, processing data, training and assessing models and conducting comparative analyses.

1.5.2 Data Collection and Preprocessing

Data were collected from seven Swedish Households(Aupke, 2023), each equipped with photovoltaic (PV) systems, encompassing a diverse range of power consumption patterns. Each dataset included time-stamped records of power usage, alongside relevant meteorological data to account for environmental influences on energy consumption. The preprocessing steps involved data cleaning, normalization, and partitioning into training and test sets, ensuring a consistent format across all datasets for fair comparison.

1.5.3 Model Development

Two primary models were developed for comparison:

1. **Federated Learning Model**: A multi-layered sequential neural network designed to forecast upper, lower, and mean levels of power consumption. The model was trained using a federated learning approach, allowing for decentralized model training across the Prosumer datasets without data centralization.
2. **Split Learning Models:** Multiple configurations of split learning models were tested, varying the distribution of neural network layers between the client and server sides. Experiment 1 allocated one layer on the client side, Experiment 2 allocated two layers, and Experiment 3 allocated three layers, to explore the impact of client-side complexity on model performance.

1.5.4 Experimental Setup

Each experiment was conducted using a dedicated split learning and federated learning framework (PySyft). The framework facilitated the training of models under different configurations, ensuring that data privacy was maintained by minimizing data exchange between clients and servers. The performance of each model was evaluated using several metrics, including Mean Quantile Loss (MQL), Mean Prediction Interval Range (MPIR), and Mean Absolute Error (MAE).

1.5.5 Model Training and Evaluation

Models were carefully trained on datasets specific to each Prosumer, emphasizing the reduction of loss metrics and enhancement of prediction accuracy. The evaluation of these models was carried out on a held-out test set, which constituted 20% of the data from each Prosumer, aimed at assessing the models' ability to generalize across unseen data. This methodological approach ensured a comprehensive understanding of each model's predictive performance and its applicability to real-world scenarios.

1.5.6 Comparative Analysis

The outcomes of the federated and split learning experiments underwent thorough analysis to benchmark their performance against various metrics. This analysis aimed to discern which learning approach—federated or split learning—excelled in terms of accuracy, computational efficiency, and compliance with data privacy norms. Furthermore, the study delved into the effects of augmenting client-side complexity within split learning models, evaluating its influence on both prediction accuracy and the models' capacity for generalization.

1.5.7 Statistical Analysis

To validate the significance of observed differences in performance metrics between the models, statistical methods were utilized. Specifically, paired t-tests were applied to compare Mean Absolute Error (MAE) values across various experimental conditions. This approach ensured the robustness and reliability of the findings, providing a solid statistical foundation for evaluating the comparative effectiveness of the models under study.

1.6 Hypothesis

Below are the hypotheses formulated to guide our investigation into the performance of split and federated learning models in smart grid power consumption forecasting. These hypotheses are grounded in preliminary research and theoretical considerations, aiming to test the models' effectiveness, efficiency, and privacy preservation capabilities within the context of smart grids.

- Hypothesis 1 suggests that split learning, due to its architecture enabling flexible computational task distribution between client and server, will excel over federated learning in power consumption forecasting. This is based on the belief that such a structure will enhance prediction accuracy, computational efficiency, and data privacy.
- Hypothesis 2 posits that enhancing the number of layers processed by the client in a split learning model will improve prediction performance. This hypothesis is rooted in the idea that processing more complex features locally on client devices will allow for the detection of subtler patterns in power consumption data, thus boosting the model's overall predictive accuracy.

These hypotheses are pivotal for investigating innovative smart grid management strategies, focusing on refining energy forecasting methods amidst considerations for privacy and computational limitations.

1.7 Outline

This thesis explores the shift towards renewable energy in smart grids and the importance of accurate power consumption forecasting.

- Chapter 2 discusses time-series forecasting and machine learning in energy forecasting, including Neural Networks, Distributed Learning Models, and Federated and Split Learning.
- Chapter 3 discusses the selection and optimization of hyperparameters critical to model performance. It also outlines the federated learning approach and the tri-model split learning framework, including their design and training processes
- Chapter 4 provides a comprehensive comparison of the federated learning experiment with the split learning experiments and highlights key takeaways.
- Chapter 5 analyzes the results, highlighting the advantages of federated and split learning in smart grid forecasting.
- Chapter 6 concludes by summarizing key findings and suggesting future research directions.

References are provided in the final section for further reading.

Chapter 2: Background and Related Work

This section lays the basis for our thesis, focusing on the domain of Time Series Forecasting and its intersection with Machine Learning, particularly Neural Networks like LSTMs. Time series forecasting is a pivotal element in predictive analytics, playing a crucial role in making accurate projections across various sectors. We'll introduce Time Series forecasting, tracing its historical evolution, and addressing its key challenges, thereby highlighting its critical importance in the modern technological landscape. This section will also explore the concepts of distributed learning, providing essential insights that contribute to a deeper understanding of Time Series forecasting's role and potential in the broader context of advanced data analysis and machine learning applications.

2.1 Time-Series Forecasting

Time-series forecasting is a vital statistical technique widely used across various industries to guide decision-making through analyzing historical data patterns. This approach is valuable in sectors such as finance, meteorology, and inventory control. Time series forecasting employs a range of methodologies, from basic linear techniques like moving averages to more

complex models like ARIMA and neural networks. These models are proficient at detecting patterns in temporal data to forecast future occurrences or trends. While time series forecasting is essential for deriving meaningful insights, it encounters obstacles like the non-stationarity of data and the complications of choosing the right model. The field has seen significant enhancements thanks to recent developments in machine learning and artificial intelligence, leading to improved precision and forecasting capabilities. The success of these forecasting methods heavily relies on the quality of the data and the appropriateness of the chosen model to the inherent patterns in the data.

2.1.1 Time-Series Evolution

The historical evolution of time series forecasting reflects significant advancements in statistical methods and computational power. The origins of time series analysis date back to the early 20th century with Yule's pioneering work on autoregressive models (Yule, 1927), followed by Wold's introduction of moving average models in the 1930s. A major milestone was the development of the ARMA model by Box and Jenkins in the 1970s, which revolutionized time series forecasting (Box, Jenkins, Reinsel, & Ljung, 2015). The late 20th century saw further advancements with the advent of computers, leading to more complex models like ARIMA and seasonal decomposition. Recently, the focus has shifted to machine learning techniques, particularly neural networks, and deep learning, which excel in handling large and complex datasets. Among these, Long Short-Term Memory (LSTM) networks (Hochreiter & Schmidhuber, 1997), stand out. LSTMs, a recurrent neural network, are adept at recognizing patterns in data sequences and are particularly effective in time series forecasting. Their ability to remember long-term dependencies makes them well-suited for predicting complex temporal sequences. These advancements have enhanced the accuracy of forecasting models and expanded their applicability in various real-world scenarios.

2.1.2 Time-Series Key Challenges

Time-series forecasting is a complex field that faces several critical challenges across different sectors, including economic and environmental sciences. An issue in this area is the inherent unpredictability and variability of time-series data, often influenced by unforeseen external factors. Forecasting models must manage uncertainty to ensure the accuracy of predictions effectively. Another significant challenge is the non-stationary nature of time-series data, where statistical properties can vary over time. This necessitates using advanced techniques, such as differencing or transformation, to stabilize the data. Seasonality is another crucial aspect, particularly in sales, weather, and energy consumption datasets, requiring models that accurately identify and predict these seasonal trends. The exponential growth of data today poses challenges for efficient computation and real-time processing. Integrating machine learning with traditional time-series methods represents a modern solution to these challenges, enhancing both accuracy and adaptability in forecasting models.

2.1.3 Components of Time-Series

In Time Series analysis, understanding the complex patterns and behaviors embedded within data is crucial for accurate forecasting and effective decision-making. Time series data is rich with information that, when decoded, can reveal valuable insights. This data is particularly prevalent in fields such as economics, finance, and environmental science, where it aids in predicting future trends based on past patterns. There are three primary components central to the analysis of time series data: Seasonal, Trend, and Random. Each of these components plays a distinct role in shaping the overall characteristics of the data. Let's explore each of these components in detail:

1. **The Seasonal Component** captures the regular, predictable patterns that recur at specific intervals over time, known as seasonality. These patterns are often linked to calendar events and can be observed in various forms, such as annual temperature cycles, quarterly sales fluctuations in retail, or daily traffic patterns. Understanding and accurately modeling the seasonal component is crucial for effective planning and forecasting in industries where seasonal factors play a significant role. For instance, energy companies depend on seasonal forecasts to predict demand fluctuations due to weather changes (Luan et al., 2020; Shiwakoti et al., 2023). The ability to anticipate these seasonal variations enables organizations to allocate resources efficiently, manage operational costs, and maintain a competitive edge. Advanced statistical methods, including seasonal decomposition of time series (STL) and seasonal ARIMA models, are commonly utilized to separate and analyze these patterns, providing a clearer understanding of their impact on the overall time series data (Dozie & Ibebuogu, 2023; Maksood & Achuthan, 2017).
2. **The Trend Component** in a time series represents the long-term movement of the data. This trend can manifest as upward, downward, or even horizontal (stable) and is key to understanding the overall direction of the data over time. In economic data, for example, a consistent upward trend often signifies a period of economic growth, whereas a downward trend might indicate a recession or economic downturn. Trends in time are affected by multiple factors, including demographic shifts, economic policies, market dynamics, and technological advancements(Hyndman & Athanasopoulos, 2018). Moreover, the identification and analysis of trends enable forecasters and analysts to make long-term predictions and strategic decisions. The methods used to identify and analyze trends vary, ranging from simple visual inspection to sophisticated statistical techniques like moving averages or linear regression models. These methods help in short-term fluctuations, providing a clearer picture of the data's trajectory over time.
3. **The Random Component**, often termed 'noise', accounts for unpredictable fluctuations that are not explained by Seasonal and Trend components. This element captures variations resulting from unexpected or short-lived occurrences, like market shifts or natural events. Its presence, although erratic, is critical in time series analysis for constructing effective forecasting models. Analysts face the challenge of differentiating

these random fluctuations from significant data trends, a crucial step for precise forecasting and strategic decision-making.

In addition to these primary components, some time series analyses also consider a **Residual Component**, which is what remains after accounting for the primary components like trend and seasonality patterns. It represents the portion of the data not explained by the model, essentially the error or noise in the series. Analyzing the residual component is crucial for assessing the effectiveness of a time series model. If the residuals display random behavior without any patterns, it suggests that the model has adequately captured the data's structure. However, if there are discernible patterns in the residuals, it may indicate that the model has missed some aspects of the data, necessitating further refinement.

To gain a comprehensive view of these components, utilizing statistical software such as R, Python, or Excel is beneficial. These applications are capable of breaking down a time series into its fundamental elements, thereby facilitating a more transparent comprehension of the inherent patterns and trends. The figure below visualizes such components:

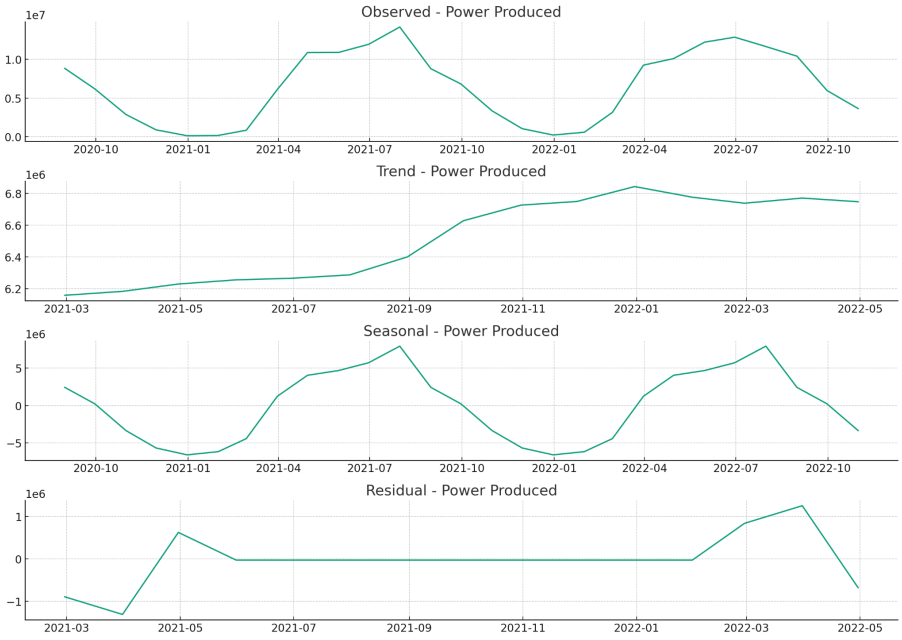


Figure 2.1: Components of Time-Series

1. **Observed - Power Produced:** Displays the actual monthly totals of power produced since August 2020.
2. **Trend - Power Produced:** Shows the underlying trend in power production, independent of seasonal effects or irregular fluctuations.
3. **Seasonal - Power Produced:** Highlights the regular, predictable seasonal patterns in power production.

4. **Residual - Power Produced:** Represents the unexplained variations or anomalies in the production data.

2.1.4 Stationarity and Seasonality

Stationarity and Seasonality are two fundamental concepts of time series data that significantly influence how the data should be analyzed and forecasted. **Stationarity** refers to a time series whose statistical properties, such as mean, variance, and autocorrelation, are constant over time. This is vital because many forecasting models assume or require the data to be stationary. Non-stationary data often lead to unreliable and spurious results in models. Techniques like differencing, transformation, or using models like ARIMA are employed to achieve stationarity.

Seasonality, on the other hand, indicates the presence of variations that occur at specific regular intervals less than a year, such as weekly, monthly, or quarterly. Seasonality can arise due to various factors like weather, holidays, or business cycles and is crucial for understanding and predicting patterns in time series data. Seasonal variations are typically handled by seasonal adjustment methods or using models that explicitly account for seasonality, such as SARIMA (Bawdekar et al., 2022).

2.2 Machine Learning in Time Series Forecasting

Machine learning has significantly advanced the field of time series forecasting, offering powerful tools to handle complex and large datasets. Unlike traditional statistical methods, machine learning models can capture non-linear relationships and interactions in time series data without requiring explicit specification of the underlying data structure; this makes machine learning particularly effective for forecasting in domains with complex dynamics. Recent developments in machine learning, especially deep learning techniques like Long Short-Term Memory (LSTM) networks, have further enhanced forecasting accuracy. These models are capable of learning from and making predictions based on large volumes of historical data. The integration of machine learning in time series forecasting represents a significant shift towards more data-driven, adaptive, and accurate predictive modeling (Masini, Medeiros, & Mendes, 2020).

2.2.1 Neural Networks and Deep Learning

Neural networks, a cornerstone of modern computational technology, are designed to mimic the human brain's complex processing capabilities. These networks comprise layers of interconnected nodes, or 'neurons,' which work in unison to analyze and interpret vast amounts of data. This structure allows neural networks to learn and adapt from experience, much like the human brain, making them proficient in handling dynamic and nonlinear relationships inherent in time series data. Deep learning, a more advanced branch of machine learning, builds upon this

foundation by employing multi-layered neural networks. These deep networks, particularly architectures like Long Short-Term Memory (LSTM) networks, capture temporal dependencies and subtle nuances in sequential data, progressively extracting higher-level features from raw input. Integrating neural networks and deep learning in time series forecasting marks a significant evolution in predictive modeling. It represents a shift towards more adaptive, data-driven, and precise methodologies, making these models more valuable for predicting future events across various domains, from financial markets to environmental changes. This advancement in neural networks and deep learning paves the way for more detailed insights and informed decision-making in our rapidly evolving world.

2.2.2 Historical Evolution: From Perceptron to Advanced Models

Neural networks originated in the 1950s with the creation of the perceptron (refer to figure 2.2) by Frank Rosenblatt at the Cornell Aeronautical Laboratory in the United States. This early model was simplistic yet groundbreaking, laying the foundation for the expansive growth in the field. Demonstrating the potential of machines to perform classification tasks, the perceptron represented the first significant step towards developing more complex neural network models.

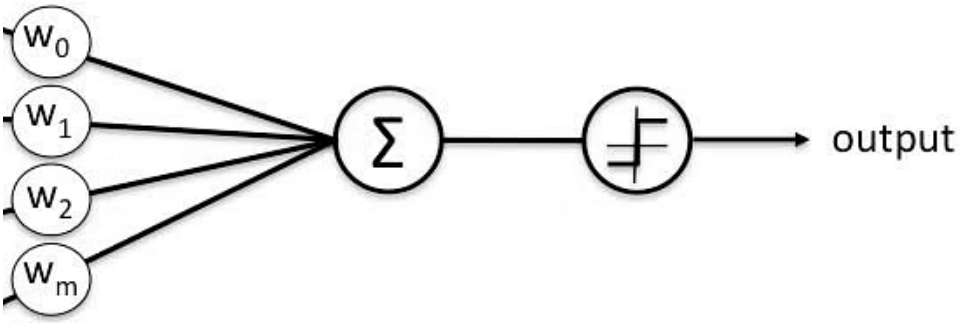


Figure 2,2: Representation of a Single Layer Perceptron (Mundkur, 2018)

In the 21st century, neural network research advanced drastically due to the dramatic increase in computational power and data availability. This progress enabled the development of sophisticated and efficient models, particularly enhancing time series forecasting where complex data analysis is crucial. Later, Deep learning, especially with the beginning of Long Short-Term Memory (LSTM) networks, emerged as a significant advancement in this domain. These networks excel in handling the intricacies of time series data, such as non-linear relationships and temporal dependencies, by processing extensive historical data. The adaptability and learning prowess of LSTM networks have made them instrumental in making accurate predictions across various domains, from financial markets to environmental studies.

As neural networks continue to evolve, they are becoming increasingly essential in deciphering the growing volumes of data, driving forward the field of predictive modeling, particularly in time series forecasting.

2.2.3 Neural Network Architecture: A Reflection of Biological Complexity

The architecture of a neural network is a testament to design precision, echoing the complexity of its biological counterpart in the human brain. At its heart lies an interconnected labyrinth of neurons or nodes, systematically organized into distinct layers, each playing a pivotal role in the network's data processing and learning mechanisms.

Input Layer serves as the gateway for data entry, each neuron in this layer represents a unique aspect of the input data. In scenarios like power production forecasting, these neurons could correspond to variables such as historical power output, weather conditions, or temporal factors(Sharma et al., 2023). **Hidden Layers** are positioned between the input and output layers, these layers form the neural network's computational nucleus. Here, neurons intricately process the input data through a series of mathematical transformations, defining the network's complexity and depth (Zhang, Yu, & Xu, 2020). **Output Layer:** This layer culminates the network's processing journey, delivering the final prediction or output, such as the forecasted power output in energy forecasting models (Her et al., 2022; Hansda & Murmu, 2023).

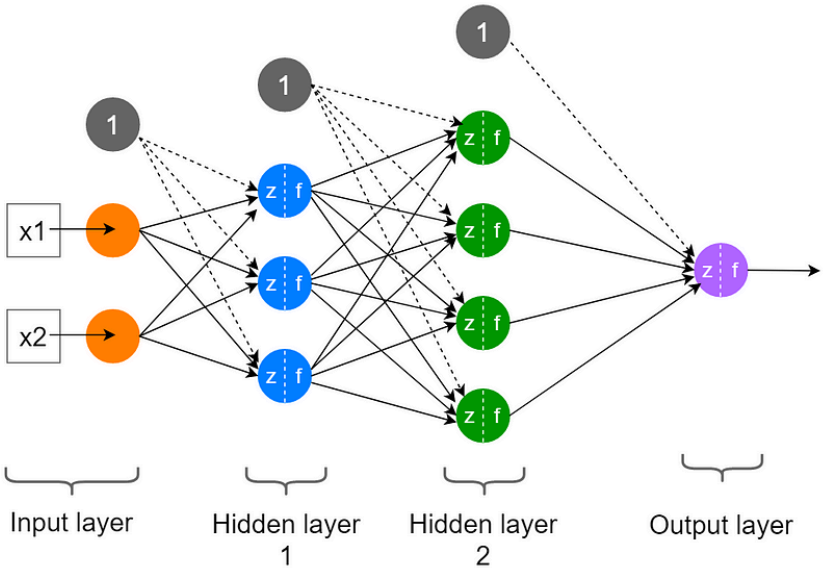


Figure 2.2a: Neural Network architecture (Pramoditha, 2022)

2.2.4 Activation Functions: The Catalysts of Learning

Integral to neural networks are the activation functions - Sigmoid, ReLU, and Tanh. These functions infuse non-linearity into the network, enabling it to learn and model complex

- Sigmoid: Maps inputs to a range between 0 and 1, ideal for binary classification and probabilistic outputs.
- ReLU (Rectified Linear Unit): A piecewise linear function that has become a staple in neural networks for its efficiency in mitigating the vanishing gradient problem.
- Tanh (Hyperbolic Tangent): Similar to Sigmoid but maps inputs between -1 and 1, suitable for modeling negative outcomes. data patterns.

These functions are pivotal in enabling neural networks to backpropagate errors and update weights, a critical aspect of their learning and generalization capabilities.

Learning Process in Neural Networks: A Symphony of Algorithms and Mathematics

The learning process in neural networks is an intricate interplay of forward propagation and backpropagation, augmented by loss functions and optimization algorithms.

- Forward Propagation: Involves data processing through the input layer, transformation in hidden layers, and output generation in the output layer.
- Backpropagation: Starts with error calculation, followed by gradient descent to adjust weights, thereby minimizing prediction errors.
- Loss Functions and Optimization Algorithms: Loss functions like MSE and Cross-Entropy quantify prediction errors, while optimization algorithms such as Gradient Descent and its variants fine-tune the network's weights. Among these, the Adam optimizer has gained prominence for its efficiency in handling sparse gradients and adapting the learning rate during training, making it a preferred choice in many neural network applications.

This dynamic and iterative learning process is fundamental to a neural network's ability to adapt, refine its understanding, and enhance predictive accuracy, making it an indispensable tool in various predictive modeling tasks, including time series forecasting in power production.

2.3 Recurrent Neural Networks (RNNs): Specialized for Sequential Data

Recurrent Neural Networks (RNNs) are specialized for sequential data analysis, ideal for time series forecasting and natural language processing, where data points are interconnected and time-sensitive. Unlike traditional neural networks with independent inputs and outputs, RNNs use previous outputs as current inputs (as seen in Figure 2.3), crucial for tasks like sentence completion. Their defining feature is the Hidden Layer or Memory State, which retains sequence information, allowing consistent processing across inputs with simplified parameter complexity. This memory aspect enables them to capture temporal dynamics in data effectively.

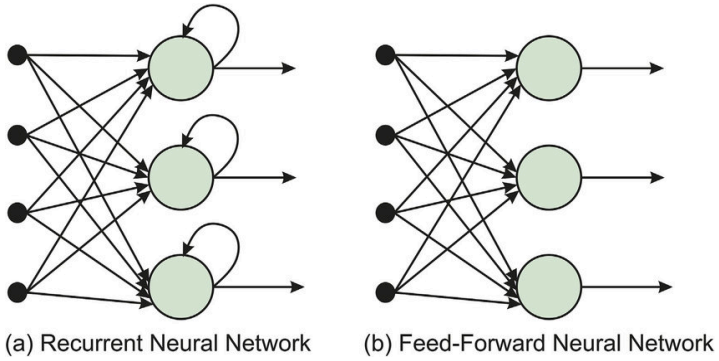


Figure 2.3: Recurrent and Feedforward Networks (Eliasy & Przychodzen, 2020)

RNNs can accurately predict future power production or market trends by utilizing past data. Their internal structure allows them to remember and learn from the historical patterns and trends in the data. For instance, RNNs can analyze past electricity demand and supply data, weather conditions, and other relevant temporal sequences to predict future power needs, making them a valuable tool for making informed predictions about future events.

2.3.1 RNN Architecture: Embracing Temporal Dynamics

RNNs are specifically designed to process sequential data with high accuracy. As we discussed earlier, these networks can effectively capture temporal dynamics, which means they can learn from past data patterns. RNNs typically consist of three essential layers, each playing a significant role in the network's overall performance and functionality, especially in tasks such as time series forecasting and language processing. The **input layer** in a recurrent neural network receives sequential data, where each neuron corresponds to a time step in the sequence. The data then moves to the **recurrent hidden layer**, where neurons process current inputs while retaining a memory of past information through their internal states. This allows the network to make informed predictions by integrating both current and historical inputs, capturing the temporal dynamics in the data. Finally, the **output layer** generates a final output, which could be a prediction for the next time step in a sequence or a classification decision depending on the task.

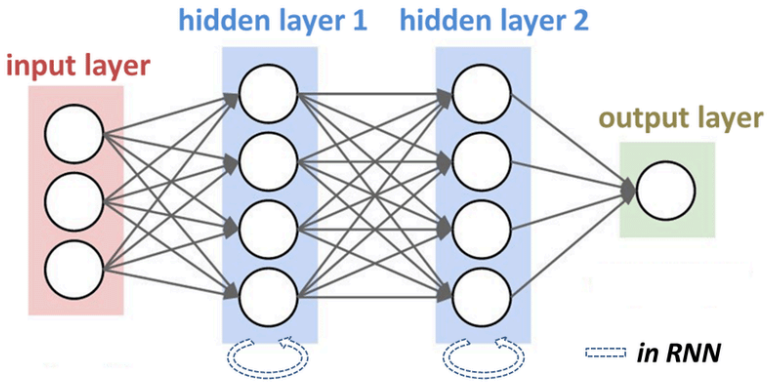


Figure 2.3.1: RNN Architecture (Ma et al., 2019)

2.3.2 Challenges with RNNs: Vanishing and Exploding Gradients

Despite their effectiveness in sequential data analysis, RNNs face challenges like vanishing and exploding gradients, particularly in long sequences. Vanishing gradients occur when the gradients become too small, hindering the network's ability to learn from earlier data points, thus affecting its capacity to understand long-range dependencies (Rehmer & Kroll, 2022). Exploding gradients, on the other hand, involve massive gradients, leading to unstable training and deviation. Solutions to these issues include using gated architectures like LSTMs and GRUs, which introduce gates that regulate the flow of information through the network, allowing them to retain important information over long sequences while discarding irrelevant data (Krishnan et al., 2021). This design mitigates the vanishing gradient problem by maintaining a more constant error flow across layers. Another approach to tackle these issues is gradient clipping, a technique where gradients are artificially limited to a maximum value to prevent them from exploding. This method ensures that the updates to the model weights remain within a manageable range, promoting more stable and consistent training. Furthermore, advanced optimization algorithms like Adam and RMSprop have been developed to adapt the learning rate during training dynamically. These optimizers help in navigating the complex loss landscapes of RNNs more effectively, reducing the impact of both vanishing and exploding gradients.

2.3.3 LSTM Networks in Time Series Forecasting: Overcoming RNN Limitations with Advanced Memory Management

Long Short-Term Memory Networks (LSTMs), a specialized variant of Recurrent Neural Networks (RNNs), are ingeniously designed to tackle the challenges of vanishing and exploding gradients, which are common in traditional RNNs. This is achieved through a sophisticated internal architecture that enables the sequential processing capabilities of RNNs while significantly enhancing the ability to capture long-range dependencies within data (Kumar et al., 2020).

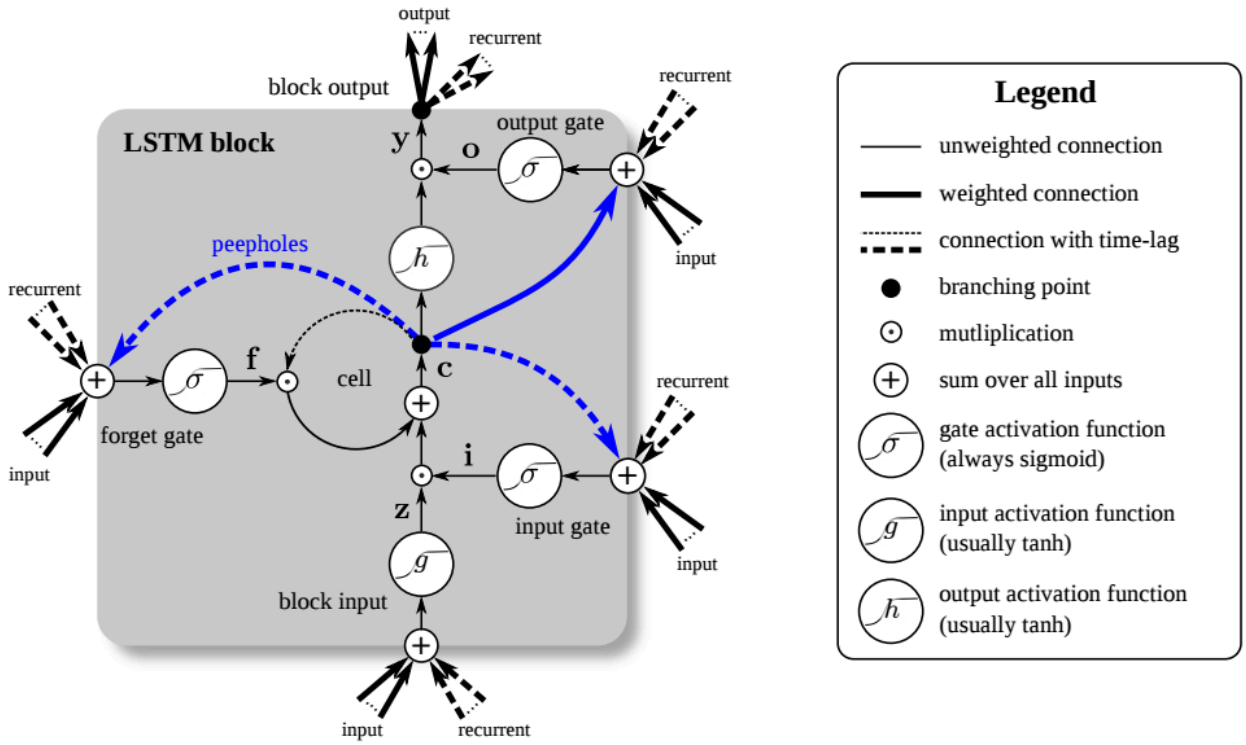


Figure 2.3.2: A Long Short-Term Memory (LSTM) unit. (Greff et al., 2015)

LSTM Architecture: At the core of LSTMs are memory blocks containing several components crucial for advanced memory management: input gates, output gates, and forget gates. These elements work in harmony to regulate the flow of information, allowing the network to selectively retain or discard data across time steps, thus providing a nuanced control over the memory cell's state.

- **Input Gate:** Manages how much of the new information is allowed into the cell, enabling the network to add to its memory.
- **Forget Gate:** Determines what information is no longer relevant to the task at hand and thus should be removed, helping to prevent unnecessary clutter in the cell's memory.
- **Output Gate**: Decides what information is useful and should be output from the cell at the current step, based on the input and the memory of the cell.

Optimization Techniques: Beyond their architectural advancements, LSTMs also leverage sophisticated optimization techniques like Adam. This optimizer enhances training efficiency and convergence by adapting learning rates for different parameters, further refining the network's learning process.

Application in Time Series Forecasting: LSTMs are particularly valuable in time series forecasting, where understanding historical data and its temporal dependencies is crucial. Whether predicting power production, financial market trends, or weather patterns, LSTMs, with their enhanced memory management capabilities, offer a significant advantage. They excel in modeling complex sequences and dependencies over time, making them a powerful tool for tasks where the historical context is integral to accurate forecasting (Wang et al., 2023).

LSTMs represent a pivotal development in the field of deep learning for sequential data analysis. Their unique ability to manage memory effectively, coupled with advanced optimization techniques, allows them to overcome the limitations of traditional RNNs, making them a cornerstone in the advancement of time series forecasting.

2.4 Distributed Deep Learning in Time Series Forecasting

Distributed Deep Learning (DDL) has become a pivotal tool in enhancing time series forecasting, particularly in managing large-scale data and computational intensity. It allocates deep learning tasks across multiple computing nodes, utilizing parallel processing to handle complex time series data efficiently. This approach not only improves scalability but also boosts computational efficiency, which is crucial for processing vast datasets that challenge traditional single-node systems due to their limited computational and memory capacities. DDL is especially effective in dealing with high-dimensional time series data, a common aspect of forecasting tasks. Central to DDL's effectiveness are architectures like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), including LSTM variants, which play a critical role in extracting detailed features from complex temporal sequences. Furthermore, DDL enables the deployment of more intricate and deeper neural network models, which might be too demanding for single-machine operations, potentially enhancing the accuracy of forecasting results. A significant aspect of DDL in time series forecasting is its support for collaborative learning while maintaining data privacy. Techniques such as Federated Learning facilitate collaborative model training across various organizations or locations without the need for actual data sharing, thus addressing privacy and regulatory concerns. With increasing data volumes and rising computational demands, DDL stands as a robust and dynamic framework, well-positioned to drive significant advancements in forecasting methodologies.

2.4.1 Data Parallelism

Parallelism in computing refers to the process of breaking down larger tasks into multiple, smaller ones to be executed simultaneously by multiple processors. This concept is

crucial in optimizing computational processes, especially in handling large and complex tasks. Parallelism can be implemented in various ways, depending on the nature of the tasks and the architecture of the system. In this section, we will go over Data and Model Parallelism. To begin with, Data Parallelism involves dividing a substantial dataset into smaller segments and processing these segments concurrently (parallel processing) across multiple processors or computing nodes which allows for the simultaneous execution of tasks, significantly enhancing efficiency and processing speed (Graser et al., 2023). When applied to time series data, Data Parallelism is exceptionally beneficial as it enables the rapid processing of these large datasets, a crucial factor in training deep learning models like Neural Networks (Tin et al., 2022). Each processor in a Data Parallelism setup works on a different subset of the data, facilitating faster model training by aggregating the learning from these subsets. This approach is particularly advantageous for real-time analysis of time series data, such as in financial markets or IoT sensor data, where swift processing is essential for timely insights and decision-making. Moreover, Data Parallelism scales effectively with the growing volume of data, maintaining processing efficiency even as data loads increase. Typically implemented in distributed computing environments, such as cloud platforms, Data Parallelism allows for dynamic resource allocation, adapting to the varying demands of time series data analysis. Thus, Data Parallelism is an indispensable technique in modern time series forecasting, enabling the handling of large datasets with increased speed and efficiency, and facilitating more effective real-time data analysis.

2.4.2 Model Parallelism

Model Parallelism offers a distinctive computational strategy, different to Data Parallelism, to manage large-scale machine learning challenges, such as those in time series forecasting. Unlike Data Parallelism, which divides a dataset among multiple processors, Model Parallelism involves partitioning the machine learning model itself. This technique distributes different components or layers of a complex model across several computational units, making it especially useful for extensive neural network models that exceed the memory capacity of a single processor (Abbas et al., 2019). In time series forecasting, where deep neural networks are often employed to decode complex temporal sequences, Model Parallelism facilitates the processing of these sizable models. It allows for the division of computational tasks, where one processor might compute the initial layers of a deep learning model, while others handle later layers. This distribution of computational load is essential in time series analysis, particularly for models with high computational requirements or those needing swift training and inference. Model Parallelism thus enables the application of advanced models in time series forecasting, which might be unfeasible in a single-processor setup due to computational limitations. By utilizing multiple processors, Model Parallelism makes it possible to train and utilize complex models, potentially enhancing the accuracy and depth of forecasting insights.

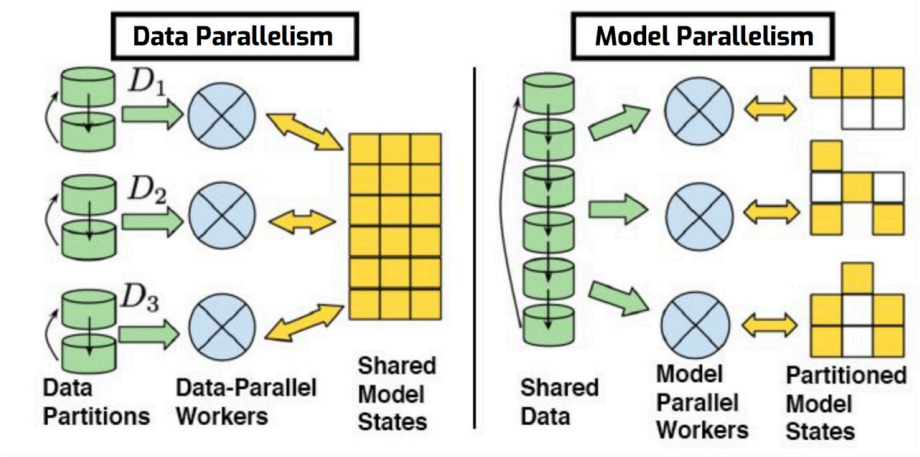


Figure 2.4: Model and data parallelism (Polosukhi, n.d)

2.5 Federated Learning in Time Series Forecasting

Federated Learning (FL) represents a paradigm shift in time series forecasting by enabling collaborative model training across multiple clients while ensuring data privacy and reducing communication costs. Unlike traditional centralized data storage approaches, FL adopts a decentralized framework where each client leverages its data locally to contribute to a collective model, coordinated by a central server. This decentralized approach ensures that raw data remains exclusively on the client side, effectively addressing privacy concerns. In FL, clients compute model updates independently, and only these updates, rather than the raw data, are communicated to the server. This methodology aligns with the principles of Data Parallelism, as previously discussed, but with a crucial difference: FL naturally enables parallel processing of data across clients without the need for explicit data partitioning, as it separates the training process from direct data access..

Federated Learning (FL) is a useful technique when dealing with data distributed across multiple devices, particularly in situations where data volume or privacy concerns are significant. However, implementing FL in time series forecasting poses unique optimization challenges that differ from typical distributed optimization problems. One such challenge is the heterogeneity in data also known as the non-independent and identically distributed (non-IID) nature. This means that each client's dataset may have distinct characteristics that do not necessarily mirror the overall population's data distribution, potentially leading to skewed model training and impacting the forecasted outcomes' accuracy and generalizability (Sun et al., 2023; Chen et al., 2023). Additionally, there is often a significant discrepancy in the quantity of data available across different clients. While some may have access to extensive datasets, others may have limited data, creating disparities in each client's contribution to the model and potentially resulting in biased or underfitting models (Qi et al., 2023). Another significant challenge is scalability and distribution. FL involves potentially a vast number of clients contributing to the model training,

and managing as well as efficiently aggregating updates from such a large cohort, especially when individual data contributions are small, poses a considerable scalability challenge (Qi et al., 2023).

Lastly, communication constraints form a critical aspect of FL, relying on client-server communication for model updates. However, clients often face limitations such as intermittent connectivity, slow network speeds, or high communication costs that can hinder the timely and efficient exchange of model updates and impact the overall effectiveness of the training process. Addressing these challenges is crucial for leveraging the full potential of FL in time series forecasting. Solutions need to be tailored to handle data heterogeneity, ensure balanced contributions from diverse data sources, scale efficiently with the number of clients, and optimize communication strategies to accommodate varying network conditions (Rosero et al., 2023).

2.5.1 Split Learning (SL) in Time Series Forecasting

Split Learning (SL) offers a transformative approach in time series forecasting, particularly in scenarios where data privacy is paramount and computational resources are distributed. Unlike traditional centralized learning systems, SL allows for the training of a shared model by splitting the learning process between the client and the server. In this setup, the raw data remains on the client side, ensuring data privacy and reducing the need for extensive data communication (Lyu et al. 2023). The learning task in SL is facilitated through a collaborative effort between multiple clients and a central server. The clients perform part of the model computation using their local data and send only the intermediate model outputs to the server. The server then completes the remaining part of the computation. This process is somewhat analogous to Data Parallelism, but with the crucial advantage that SL inherently maintains data privacy by design, as the raw data never leaves the client's premises.

One of the main advantages of split learning is its computational and communication efficiency, especially for clients. Since only the initial layers of the neural network are trained on the client side, the computational burden is significantly reduced. However, to make SL practically viable, several unique challenges within this framework need to be addressed:

1. Data Privacy and Security: Ensuring that the intermediate data shared between the client and the server does not compromise the privacy of the raw data.
2. Computational Balance: Managing the computational workload between the client and the server to optimize resource utilization and prevent bottlenecks (Chang et al., 2023).
3. Scalability: Effectively scaling the SL framework to accommodate a large number of clients with varying computational capabilities.
4. Communication Efficiency: Minimizing the communication overhead between clients and the server, is especially crucial for clients with limited or costly connectivity.

Addressing these challenges is key to leveraging the full potential of Split Learning in time series forecasting. Solutions need to focus on enhancing data security, optimizing computational workload distribution, improving scalability, and ensuring efficient communication protocols.

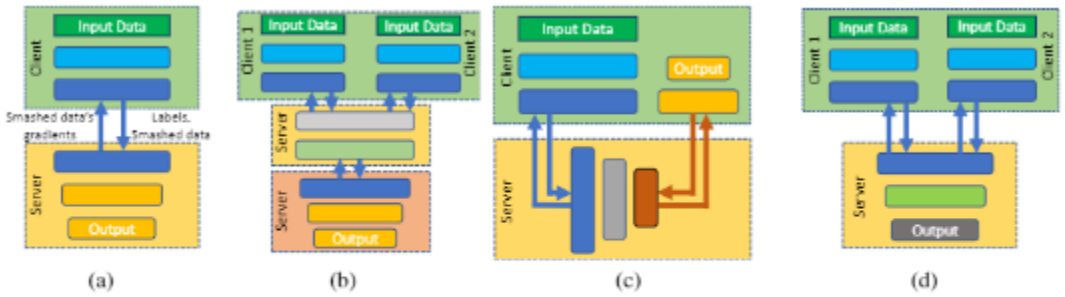


Figure 2.5.1 Split Learning Configuration (a) Simple Vanilla, (b) Extended Vanilla, (c) Without Label Sharing, and (d) Vertically Partitioned Data (Thapa et al., 2020)

Split Learning (SL) offers a versatile framework for privacy-preserving machine learning, demonstrated through various configurations each tailored to specific privacy and computational requirements. In its simplest form, Vanilla Split Learning, a client trains a model on local data and sends intermediate representations, known as "smashed data," along with labels to a server for further processing. This configuration, depicted in Figure 2.5.1a, enables a straightforward collaborative learning process without compromising raw data privacy (Yang et al., 2022). The Extended Vanilla Split Learning approach, illustrated in Figure 2.5.1b, builds upon this by permitting some processing of intermediate layers on the client side before transmission to the server, thereby reducing the server's computational burden and introducing greater flexibility in model training (Mugunthan et al., 2021). In scenarios where labels are sensitive, Split Learning Without Label Sharing offers a solution by having the client share only the smashed data, omitting labels. As shown in Figure 2.5.1c, the server processes this data up to a predetermined layer and returns the activations to the client, which then completes the forward propagation and initiates backpropagation, establishing a U-shaped information flow (Ezzeddine et al., 2023). For situations involving Vertically Partitioned Data, as represented in Figure 2.5.1d, SL adapts to cases where clients possess different features for the same samples. Each client processes its data segment and forwards the smashed data to the server, which then amalgamates these inputs for continued processing. This model is especially beneficial in multi-disciplinary studies involving diverse data types across subjects (Liu et al., 2023). Collectively, these configurations underscore SL's adaptability in meeting the diverse needs of distributed learning environments, allowing for customized data splitting and training process flows to uphold both privacy and computational efficiency.

2.6 Predictive modeling: statistical techniques and metrics

We'll cover predictive modeling and performance metrics in our analytical framework. Our approach is centered around the application of Quantile Regression, which enables us to explore the conditional distribution of power consumption across various quantiles. To evaluate the accuracy and reliability of our predictions, we utilize three metrics: Mean Quantile Loss (MQL), Mean Absolute Error (MAE), and Mean Prediction Interval Analysis. Each of these metrics provides a unique perspective on the performance of our models, allowing us to make a comprehensive assessment of their predictive capabilities. This section aims to explain the methodology and tools that form the basis of our analysis, paving the way for a detailed exploration of our findings and their implications for power consumption forecasting.

Quantile regression

Quantile regression emerges as a critical analytical tool in our study, distinguishing itself from traditional mean regression by estimating the conditional median and various other quantiles of the response variable. This approach broadens our understanding of the entire distribution of the dependent variable, which is particularly pertinent for forecasting power consumption. Unlike ordinary least squares regression that targets the mean, quantile regression excels in outlining the relationship between predictors and the response variable across the entire distribution, from the lower to the upper quantiles. This capacity is invaluable for energy consumption analysis, as it allows for the investigation of consumption patterns that span typical usage to extreme conditions, such as surges in power demand. The strength of quantile regression lies in its unique loss function,

$$L(y, \hat{y}_p; p) = \max(p(y - \hat{y}_p), (1 - p)(\hat{y}_p - y))$$

where **p** represents the targeted quantile. This loss function is adept at minimizing discrepancies across specified quantiles, thereby furnishing a detailed portrayal of energy consumption patterns. By integrating this technique, we aim to delve into the complex interplay between predictor variables and residential power consumption across the entire distribution, with a keen focus on the tails where extreme values are prominent. This methodological approach not only bolsters our neural network models with enhanced predictive accuracy but also provides a more layered understanding of power consumption patterns. Ultimately, our use of quantile regression aims to facilitate more effective energy management and informed policymaking by forecasting energy demands with greater precision, accounting for both typical and atypical usage patterns.

Mean Quantile Loss

The Mean Quantile Loss (MQL) is a metric in quantile regression analysis, designed to evaluate the model's performance by averaging the quantile loss across all observations. This metric captures the essence of quantile regression, which is to accurately predict conditional quantiles of the response variable. The formulation of the Mean Quantile Loss is given by:

$$MQL_{\Sigma} = \frac{1}{N}\sum_{i=1}^{N} L_{\Sigma}\left(y_i, \hat{\boldsymbol{y}}_i\right)$$

Where:

- N is the total number of observations,
- Yi represents the actual value of the ith observation
- ŷi is the predicted quantile value for the ith observation

Particularly valuable in energy consumption forecasting, MQL quantifies the discrepancy between observed and predicted quantile values into a singular, interpretable figure. By aggregating individual quantile losses, MQL offers a comprehensive measure that reflects the model's capability to capture the distribution of the response variable across specified quantiles accurately. Incorporating MQL into analytical frameworks not only enhances the accuracy of quantile predictions but also serves as a guide for model refinement, highlighting divergences between predictions and actual observations, especially in the distribution's tails where extreme values are prevalent. Optimizing MQL ensures that predictive models accurately reflect both the central tendency and the variability and extremities of the distribution, crucial for effective decision-making and resource management in energy consumption forecasting. This nuanced understanding of the distribution's full range and variability, facilitated by MQL, significantly influences the reliability and utility of predictive models in practical applications.

Mean Prediction Interval Range (MPIR)

The Mean Prediction Interval Range (MPIR) is a statistical metric utilized to quantify the average width of prediction intervals across a set of forecasts, it provides insights into the uncertainty or variability associated with predictions. The MPIR captures the average range between upper and lower quantile predictions across all observations to calculate the spread or uncertainty of predictions. The MPIR is defined as follows:

$$MPIR = 1/N\sum_{i=1}^{N} \hat{y}_i^u - \hat{y}_i^l$$

Where:

- N is the total number of observations,
- Ŷui represents the upper bound prediction for the ith observation
- Ŷli denotes the lower bound prediction for the ith observation

This metric ascribes the breadth of prediction intervals, where a smaller MPIR indicates tighter intervals and higher confidence in predictions, and a larger MPIR signifies greater uncertainty. Particularly in fields like energy consumption forecasting, MPIR proves invaluable by enabling analysts and decision-makers to evaluate forecast reliability through the lens of prediction uncertainty. For example, a narrower MPIR in future power demand forecasts suggests a high confidence level, aiding in more accurate planning and resource allocation. Furthermore, MPIR facilitates model refinement and selection by identifying models that achieve an optimal balance between accuracy and reliability—narrow enough to ensure precision yet wide enough to encompass actual outcomes. By integrating MPIR with other performance metrics, modelers can fine-tune their predictive models to not only achieve accuracy but also convey meaningful uncertainty estimates, essential for informed risk assessment and decision-making across various sectors.

Mean Absolute Error (MAE)

Mean Absolute Error (MAE) evaluates the accuracy of predictive models by quantifying the average magnitude of errors between predicted values and actual values, without considering their direction. Renowned for its simplicity and interpretability, MAE directly reflects the average error in the same units as the data, offering a straightforward measure of a model's prediction accuracy. The formula for calculating MAE is given:

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i|$$

A lower MAE signifies closer predictions to actual data, indicating higher accuracy, while a higher MAE points to larger discrepancies and lower accuracy. Particularly useful in applications like energy consumption forecasting, MAE provides stakeholders with a straightforward metric to assess model effectiveness and compare various models or forecasting methods. This aids in selecting the most accurate model for specific needs. Additionally, MAE's robustness to outliers ensures a balanced performance assessment across different scenarios, making it a versatile and essential metric in predictive modeling and analysis. In practical applications, such as forecasting energy consumption, MAE offers a clear and immediate understanding of the model's performance. For stakeholders, having a metric like MAE is invaluable for assessing the model's practical utility in predicting real-world outcomes. It facilitates the direct comparison of different models of forecasting techniques, enabling decision-makers to choose the model that best meets their accuracy requirements.

2.7 Related Research

In the concluding section of this chapter, we explore related research and studies that share similarities with this study. This overview serves to place our findings within the wider field, showing how our research fits into the existing body of knowledge. By looking at these related efforts, we aim to highlight the unique aspects of our study and understand the broader implications of our results. This comparison not only recognizes the work of others but also opens up possibilities for future research and collaboration.

Prediction of Solar Energy Yield Based on Artificial Intelligence Techniques for the Ha'il Region, Saudi Arabia (Kolsi et al., 2023)

This research paper published in Sustainability Journal, conducts a detailed analysis of artificial intelligence models to forecast solar energy yield. This study addresses the challenge posed by solar energy's variability, which is influenced by unpredictable climatic and geographic conditions, by utilizing daily data to assess the performance of seven AI models: Naïve (N), Simple Average (SA), Simple Moving Average (SMA), Nonlinear Auto-Regressive (NAR), Support Vector Machine (SVM), Gaussian Process Regression (GPR), and Neural Network (NN). The evaluation of these models was conducted using Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE), with findings indicating a slight preference for the Naïve and Simple Moving Average models due to their superior performance. The outcomes of this research hold significant value for decision-makers and specialists in the solar energy sector, offering crucial insights into solar system power yields and aiding in estimating the photovoltaic project payback and efficiency. This study contributes to advancing renewable energy research by enhancing the precision of solar energy predictions, thereby promoting the efficient and sustainable utilization of solar resources.

Hybrid Federated Learning Framework Based on XGBoost for Distributed Power Prediction (Liu et al., 2022)

This study introduces a groundbreaking hybrid federated learning framework that integrates XGBoost to tackle distributed power prediction challenges. This innovative model merges the strengths of both horizontal and vertical federated learning to effectively address issues related to data fragmentation. By employing boosted trees, the framework not only enhances the accuracy and interpretability of power prediction models but also introduces a dynamic task allocation scheme that ensures fairness and efficiency in the learning process. This approach marks a notable advancement in the use of federated learning within power systems, showcasing its potential to improve distributed power prediction through increased model accuracy and transparent, equitable task distribution.

Privacy-preserving Power Load Prediction Using Federated Learning (Huang et al., 2022)

Kangqian Huang and collaborators present a novel approach to power load forecasting that emphasizes privacy protection through federated learning combined with RSA-AES encryption. This methodology is specifically designed to safeguard the confidentiality of data gathered by smart meters, showcasing the capability of federated learning to facilitate collaborative data analysis while maintaining the privacy of user information. The study underscores the significance of federated learning in the energy sector, particularly for applications where the protection of sensitive data is paramount. By integrating federated learning with a robust encryption system, this research demonstrates a practical solution for conducting joint data mining on smart meter data without exposing user privacy, highlighting the method's relevance and applicability in ensuring data confidentiality in power load prediction.

Privacy-Preserving Power Consumption Prediction Based on Federated Learning with Cross-Entity Data (Liu, Zhang, Shen, & Sun, 2022)

This paper introduces a systematic privacy-preserving federated learning framework for the power system sector, facilitating collaborative learning of power consumption patterns with an innovative use of horizontal and vertical federated learning and a Diffie-Hellman-based encryption scheme for secure and efficient model training. Further explored, the framework exemplifies federated learning's potential for cross-organizational collaboration in analyzing power consumption trends while maintaining strict privacy standards. Highlighting the role of AI and federated learning in advancing energy forecasting, these studies address predictive accuracy and data privacy challenges, laying the groundwork for future research and applications in the energy sector. Additionally, "LSTMSPLIT: Effective SPLIT Learning based LSTM on Sequential Time-Series Data"(Jiang et al., 2022) presents a novel approach that combines Split Learning and LSTM networks for privacy-preserving classification of time-series data, employing Differential Privacy to enhance privacy assurance, showcasing promising avenues for secure and accurate time-series data analysis.

Chapter 3: Design, Data, and Implementation

In this chapter, we will expand upon the fundamentals of neural networks and delve into their practical application for residential power consumption forecasting. We will emphasize the effectiveness of neural networks in predicting energy usage For prosumers and provide insights into the dataset used for this purpose. Our goal is to bridge theoretical knowledge with practical application, offering a complete guide to understanding and applying neural network-based predictions in residential power consumption.

3.1 Neural Network in Power Consumption

Neural networks have emerged as a powerful tool for predicting power consumption, particularly in residential settings. A dataset encompassing various attributes such as energy bought, produced, sold, total consumed, along with environmental factors like temperature, dew point, humidity, precipitation, wind direction and speed, air pressure, and global radiation, is adequate for neural networks to model and forecast total power consumption effectively. Because neural networks can distinguish complex patterns and relationships within large datasets, they are ideal for this application. Zhang et al. (2020) showcased the effectiveness of deep learning models in predicting residential power consumption with similar datasets, emphasizing the superior performance of neural networks in handling non-linear and high-dimensional energy data compared to traditional statistical methods. Furthermore, the integration of environmental variables into the neural network model is crucial, as these factors significantly influence energy consumption patterns. Research done by Li (2019) showed that incorporating weather-related variables like temperature and humidity improves the prediction accuracy of neural network models in energy consumption forecasting. By analyzing a comprehensive set of variables, including both energy usage and environmental factors, these models can provide valuable insights for energy management and planning.

3.2 Dataset and Practical Findings

The practical work conducted in this study aims to investigate the efficacy of split learning for power consumption prediction. This research utilizes a dataset derived from Aupke's 2023 study, focusing on seven distinct Prosumers in Uppsala, Sweden, as illustrated in Figure 3.3. Building upon Aupke's foundational work, this thesis aims to extend and apply the insights gained to explore the efficacy of federated and split learning models in smart grid energy forecasting (Aupke, 2023). This dataset was collected over 14 months and includes hourly measurements.

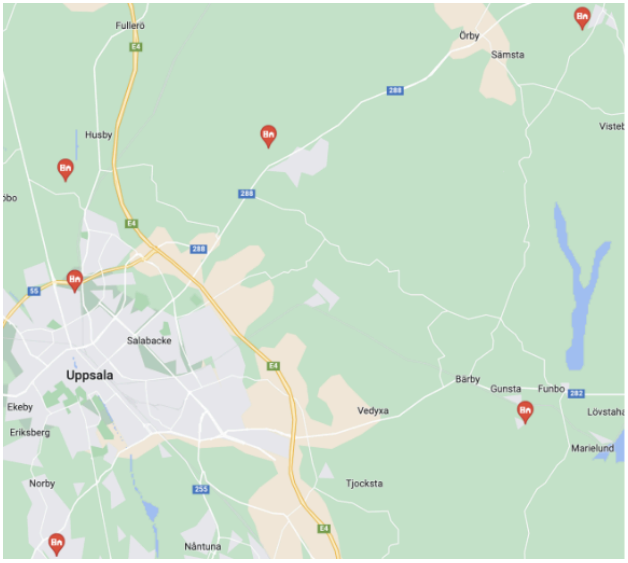


Figure 3.2a: Map of Uppsala where the houses inside our smart grid are located (Aupke, 2023)

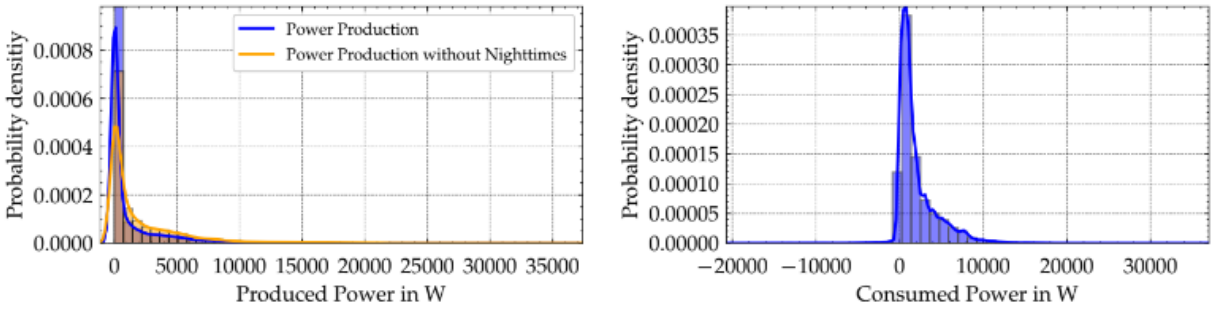
Our forecasting algorithm is trained and evaluated using a dataset labeled $D = (X, \bar{y})$, which comprises a feature matrix $\bar{X}$ with dimensions $N \times k$. Here, N indicates the frequency of data collection, and k represents the total number of features within the dataset. Each feature vector, denoted as $xi = (xi,1, ..., xi,k)^T$, is a component of $\bar{X}$. To augment the accuracy of our machine learning model in predicting PV production and consumption, we also integrate hourly weather data sourced from the Swedish Meteorological and Hydrological Institute (SMHI), as detailed in Table 3.3a below. This inclusion of external weather data aims to provide a more comprehensive basis for our algorithm's training and evaluation process.

Data Type	Feature	Unit
Weather Information	Temperature	C
	Dew Point	C
	Humidity	Percentage
	Precepitation	L/m^2
	Wind Direction	Degrees
	Wind Speed	m/s
	Air Pressure	mBar
	Global Radiation	W/m^2
Prosumer Information	Bought Power	W
	Produced Power	W
	Sold Power	W
	Consumed Power	W

Table 3.2a: Prosumer and Winter Features (Aupke, 2023)

For the machine learning models, the input features ($\bar{X}$) undergo normalization. The dependent variable, $y = (y1, ..., yN)$, reflects the power either produced (PProduced) or consumed (PConsumed) by each Prosumer. PConsumed is determined by adding the power imported from the main energy grid to the power generated by the Prosumer's PV system, then

subtracting the power exported back to the grid, formulated as PConsumed = PImport + PProduced − PExport. Notably, PConsumed may occasionally be negative within the dataset, indicating instances where a Prosumer's energy production or purchase is zero, and it solely sells stored energy from its battery to the main grid. Visual representations of the data distribution for both produced and consumed power are provided through histograms and Kernel Density Estimations (KDE) in Figure 3.3b, offering a detailed view of the data characteristics.



(a) Histogram - Produced Power

(b) Histogram - Consumed Power

Figure 3.3b shows histograms and Kernel Density Estimations (KDE) (Aupke, 2023)

Additionally, night times, during which PV cells do not generate electricity, are excluded from the production data during training and evaluation. This dataset and its characteristics provide a robust foundation for assessing the performance of split learning in power prediction.

3.3 Hyperparameters in Neural Network Modeling

Hyperparameters are vital in neural network modeling, particularly for applications like power consumption prediction. These external configuration variables are predetermined, not learned from the data, and significantly influence the learning process and model performance. This section provides a comprehensive overview of various hyperparameters, their definitions, significance, and impact. There are several hyperparameters, some of which include:-

1. **Network Architecture Hyperparameters**
   Layers and Neurons:
   - *Overview*: The architecture of a neural network is defined by its layers and the neurons within those layers. The configuration, in terms of depth (number of layers) and width (number of neurons), is crucial in dictating the network's learning capacity.
   - *Implications*: The chosen structure has a balance between complexity and simplicity. While more layers and neurons can model more intricate patterns, they also raise the possibility of overfitting. Contrarily, too few layers, and neurons might lead to underfitting, where the network fails to capture essential data characteristics.

2. **Activation Functions**
   ReLU, Sigmoid, and Tanh:
   - *Overview*: Activation functions like ReLU (defined as f(x) = max(0, x)), Sigmoid (f(x) = 1 / (1 + e^(-x))), and Tanh (f(x) = tanh(x)) introduce non-linearity into neural networks.
   - *Implications*: These functions enable the network to learn and represent complex data relationships, each having specific advantages. ReLU is particularly efficient for hidden layers, while Sigmoid and Tanh are better in output layers for binary classification tasks.
3. **Data Preprocessing Hyperparameters**
   Scaling Methods (StandardScaler, MinMaxScaler, RobustScaler):
   - *Overview*: Normalization and scaling methods adjust the input features to a common scale, improving the stability and efficiency of the learning process.
   - *Implications*: Choosing a scaling method can profoundly impact the model's performance, particularly in datasets with features of varying scales or significant outliers. For example, the RobustScaler is particularly effective in datasets where outliers might skew the feature scaling process. Additionally, the MinMaxScaler is highly effective when the data needs to be scaled within a specific range, such as between 0 and 1, which is useful for algorithms that are sensitive to the scale of the data.
4. **Training Hyperparameters**
   Optimizers (Adam, SGD, RMSprop):
   - *Overview*: The optimizer you choose, be it Adam, SGD, or RMSprop, plays a critical role in updating the network's weights during training. Each optimizer has its own set of characteristics and ways of functioning.
   - *Implications*: The efficiency of training and the quality of the model relies heavily on the chosen optimizer. Adam, for instance, is recognized for its adaptability and efficiency in handling sparse gradients, while SGD is simpler but may require more nuanced tuning.
5. **Learning Rate:**
   - *Overview*: This parameter controls the step size at each iteration during the optimization process.
   - *Implications*: An optimal learning rate is necessary for efficient training; it ensures that the model converges to a solution without overshooting or excessively prolonging the learning phase.

6. **Loss Functions (MSELoss, CrossEntropyLoss):**
   - *Overview*: Loss functions, such as MSELoss for regression and CrossEntropyLoss for classification, measure the discrepancy between predicted and actual values.
   - *Implications*: They guide the model training, aiming to minimize this difference, and are hence pivotal in shaping the model's accuracy and predictive ability.

7. **Epochs:**
    - *Overview*: An epoch represents a complete pass through the entire training dataset.
    - *Implications*: The number of epochs is a balancing act between ensuring sufficient model training and avoiding overfitting.

Choosing and adjusting hyperparameters is essential in neural network modeling, especially for complex tasks like predicting power consumption. Each hyperparameter plays a distinct role in enhancing the model's learning efficiency and overall effectiveness. The ideal setup of these parameters depends on the dataset's unique characteristics and the prediction task's complexity. Therefore, a deep understanding of these parameters is key to creating powerful and efficient neural network models.

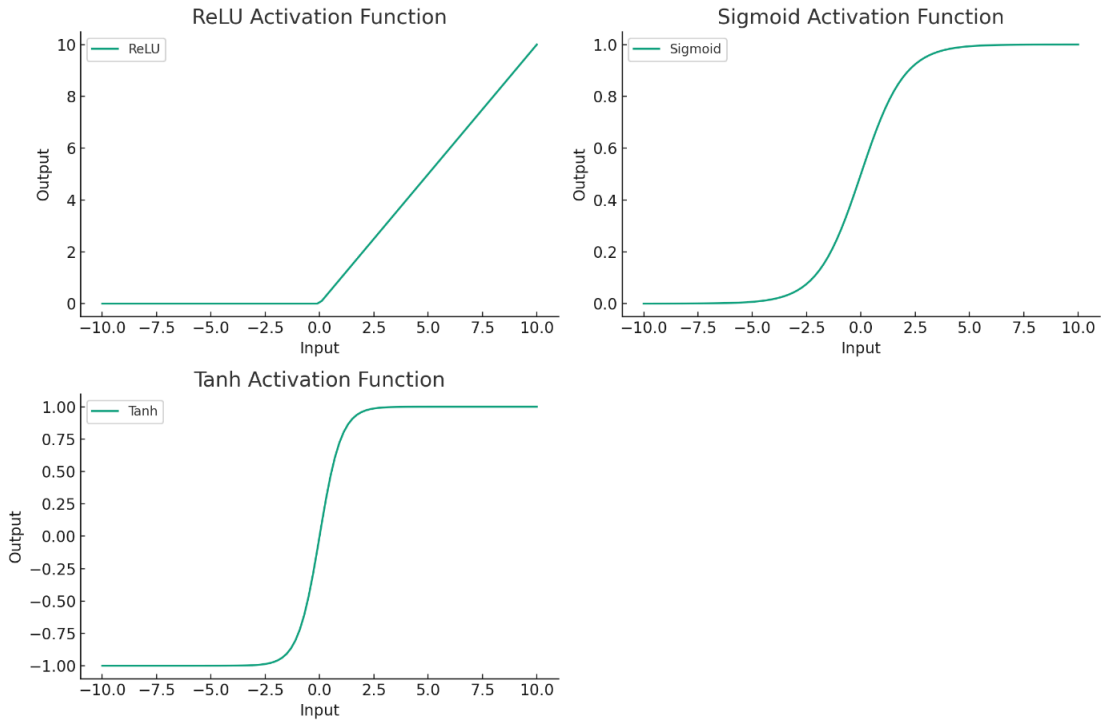


Figure 3.3a: Three commonly used activation functions in neural networks

The graphs above (Figure 3.4a) provide a visual interpretation of how various activation functions in neural networks process input values and generate corresponding outputs. This visualization is essential for grasping their functional roles within neural network models.

1. ReLU (Rectified Linear Unit): This graph shows the ReLU function, which outputs the input directly if it is positive; otherwise, it outputs zero. It's evident from the graph that ReLU maintains positive values as is and turns negative values to zero.
2. Sigmoid: The Sigmoid function graph illustrates how it transforms the input into a value between 0 and 1. This function is useful for binary classification problems.

3. Tanh (Hyperbolic Tangent): The Tanh function graph demonstrates its output range from -1 to 1. This characteristic makes Tanh a viable choice for situations where the model needs to handle negative values more effectively than the Sigmoid function.

3.4 Hyperparameter Optimisation

Grid search is a commonly used technique for hyperparameter optimization in machine learning models. This method methodically explores various combinations of hyperparameter values to identify the most effective set for optimal model performance. The key advantage of grid search lies in its exhaustive examination of the parameter space, which is vital for enhancing both the accuracy and efficiency of the model. By adopting this approach, a thorough assessment of different parameter configurations is conducted, leading to significant improvements in model outcomes, particularly in complex situations like handling imbalanced data sets or applying diverse machine learning algorithms.

In the research above (section 3.3) the grid search methodology was used to adjust the hyperparameters of our neural network model. This process played a pivotal role in enhancing our model's accuracy and overall performance. By carefully tuning the hyperparameters, we could tailor the model to the specific requirements of our task, resulting in exceptional outcomes. The following key hyperparameters were methodically varied and evaluated:

Hyperparameter	Description
Learning Rate	In our trials, we tested learning rates from 0.1 to 0.0001 and found that a rate of 0.001 was optimal. This learning rate provided a good balance between quick convergence and stability throughout the learning phase.
Batch Size	Our experiments involved testing batch sizes of 32, 64, 128, and 256. The optimal batch size was determined based on the memory capacity of our system and the efficiency of the training process.
Number of Epochs	We considered various epoch numbers, including 100, 300, and 500. The selected number allows effective learning from the training data while avoiding overfitting (Kale et al., 2021).
Optimizer	Upon evaluating various optimizers including SGD and RMSprop, the Adam optimizer was selected due to its quicker convergence and greater overall efficiency.
Layer Sizes	The number of neurons in each hidden layer was adjusted, and the final architecture was selected to offer a good compromise between model complexity and computational efficiency.

Activation Functions	We evaluated different functions, including Leaky ReLU and Sigmoid. ReLU was ultimately selected for its ability to prevent the vanishing gradient problem and to accelerate training.

Table 3.4: Evaluated Hyperparameters

We evaluated different hyperparameter combinations on our training data, assessing model performance by the mean squared error (MSE) on a validation set. The combination with the lowest MSE was chosen for our final model, effectively optimizing the neural network for accurate and efficient power consumption predictions.

3.5 Federated learning

The design of the federated learning system in this thesis addresses one of the most pressing issues in modern machine learning: data privacy. In traditional centralized learning systems, all data must be aggregated on a central server, posing significant privacy risks and logistical challenges (Li, Lin, Shang, & Wu, 2023). To avoid these concerns, each client in the system trains a neural network model on their data and never shares this with the server or other clients; this ensures that sensitive information remains within the confines of each client's local environment, significantly enhancing data security and user privacy (Makhija, Han, Ho, & Ghosh, 2022). Using neural networks in a federated context introduces unique challenges and considerations. One of the primary concerns is ensuring that the model trained locally on each client can generalize well when aggregated into the global model; here, choosing the neural network architecture and hyperparameters becomes crucial. Each client's neural network must be capable of capturing the underlying patterns in their local data while being flexible enough to integrate insights from other clients' models during the aggregation phase.

Utilizing PySyft, a specialized open-source framework for Federated Learning and ensuring data privacy, executes confidential Deep Learning operations. In this setup, each client's neural network model begins with standardized parameters. During the training phase, these models undergo local modifications based on the individual data of each client. Subsequently, these adjusted parameters, which include the models' weights and biases, are transmitted to a central server. This server is crucial in the learning mechanism, as it incorporates these updates. It employs federated learning techniques such as FedAvg, which integrates the updated parameters from each client's model into a collective, enhanced global model. This refined global model is then redistributed back to the clients for subsequent rounds of training.

At each client site, we implement three unique neural network models, each independently trained on the client's data. These models are specifically designed to predict distinct data aspects: the upper quantile, the lower quantile, and the mean.

1. Upper Quantile Model: Focuses on predicting the higher range of the data distribution. It is particularly useful for identifying and understanding peak values or anomalies in the dataset. By accurately predicting the upper quantile, this model provided crucial insights into extreme conditions or outlier events (Klemm, Gabriel, & Sill Torres, 2023).
2. Lower Quantile Model: Specializes in predicting the lower end of the data distribution. The model recognizes minimal or baseline levels in the dataset, offering valuable information for understanding the lower bounds of normal behavior or usage patterns (Yu et al., 2023).
3. Mean Quantile Model: Aim to predict the mean or average values of the dataset. This model plays a pivotal role in providing a general overview and a central tendency measure of the data, which are essential for routine analysis and forecasting (Gusev, Chervyakov, Alexeenko, & Nikulchev, 2023).

By focusing each model on a specific data aspect, we ensure a thorough capture of the unique characteristics and patterns in that segment. This approach leads to more accurate and nuanced predictions, as each model brings its specialized insight to a different segment of the data.

3.6 Tri-Model Split Learning for Energy Forecasting

Following a thorough exploration of the tri-model federated learning approach, we now present tri-model split learning as another framework designed to advance residential power consumption forecasting. This strategy deploys three distinct predictive models for each client, meticulously targeting the upper quantile for peak usage insights, the lower quantile for dormant consumption periods, and the mean for general usage trends. This tri-faceted methodology gives a detailed and comprehensive portrayal of energy utilization across varied Prosumer scenarios.

By integrating this tri-model approach within a split learning model, we aspire to not only chart the habitual energy consumption patterns but also to shed light on the outliers and anomalies, thereby enabling a more informed energy management system. This framework also utilizes the PySyft library, akin to our federated learning probe, this strategy is implemented through three targeted models: the upper quantile, the lower quantile, and the mean. Each model analyzes the spectrum of energy usage within a Prosumer, from peak demands to baseline efficiency and average consumption.

3.6.1 Detailed Design and Training Process

The training process for the tri-model split learning framework is a structured sequence designed to leverage local computations and centralized server capabilities, creating a synergy that maintains privacy while enhancing predictive performance. Here is an overview of the key steps involved in this process:

1. **Local Processing:**
    - Clients start by training three separate models locally. Each model focuses on a different aspect of power consumption: one for high usage periods (upper quantile), one for low usage periods (lower quantile), and one for the average usage (mean) (Liu, Zhang, Shen, & Sun, 2022).
    - After training, clients send their models' initial predictions to the server. This data includes insights into various consumption levels but does not expose any raw data, ensuring privacy is maintained.
2. **Centralized Aggregation and Computation:**
    - In this stage, the server collects outputs from each client's models, categorizing them into three types: upper quantile (high usage), lower quantile (low usage), and mean (average usage). The server then performs forward propagation, utilizing algorithms to process and interpret the gathered predictions.
    - Finally, the server aggregates and analyzes this information, formulating final predictions that encompass a comprehensive understanding of power usage across different clients and timeframes.
3. **Backpropagation and Update:**
    - The server calculates the loss for each model type using specific loss functions. These functions are tailored to suit the unique characteristics of each quantile's predictions.
    - After calculating the loss, the server computes gradients, which highlights how the models should be adjusted. These gradients are then sent back to each client. This process is done securely, making sure that no sensitive data from the clients is revealed.
4. **Local Updates:**
    - When clients receive the gradients from the server, they use this information to update the parameters of their respective models. This step involves adjusting the models based on the feedback received.
    - This process presents distributed learning, where each model is fine-tuned not just based on the client's data, but also incorporating insights gained from the collective analysis. This allows for a more comprehensive and informed learning process, tailored to each client's unique energy usage profile.
5. **Iterative Improvement:**
    - Training is a repetitive process, involving multiple rounds of refinement across various epochs. Each update involves refining the models based on the server's feedback.

- Regular validation checks are performed to evaluate model accuracy and prevent overfitting, ensuring reliable and generalizable predictions.

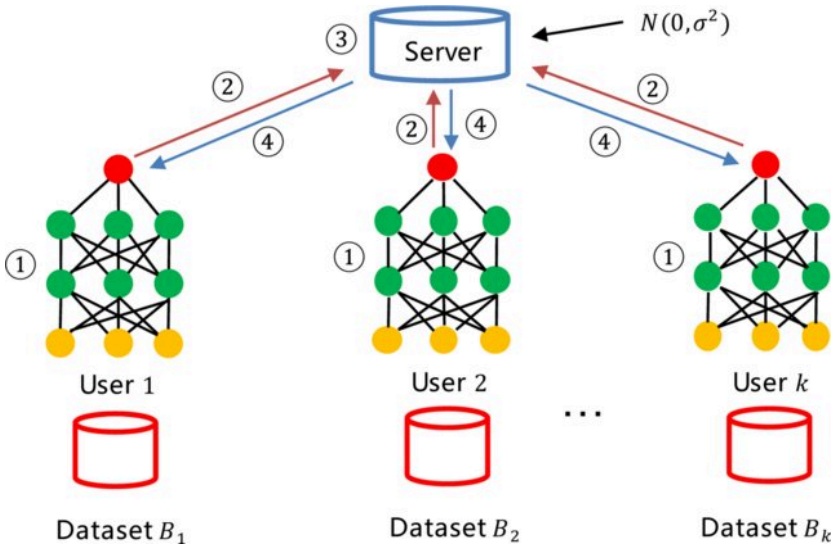


Figure 3.6.1: (Huang et al., 2020) A visual representation of the above Split Learning Training Process

Through its training process, the tri-model split learning framework achieves precise power consumption predictions while fundamentally upholding the principle of data privacy. The system's design ensures that clients' raw data remains local, safeguarding confidentiality and privacy. This approach is crucial in protecting sensitive information. Additionally, by minimizing the data exchanged between clients and the server, the architecture significantly reduces network load, thus enhancing overall efficiency.

3.6.2 Evaluation and Insights

The Mean Absolute Error (MAE) assesses the accuracy of the mean model's predictions, quantifying the average magnitude of prediction errors. For the upper and lower model predictions, the evaluation employs the Mean Prediction Interval Range (MPIR) and Mean Quantile Loss (MQL). MPIR measures the average range of the prediction intervals, providing insights into the uncertainty or variability of these predictions. MQL, on the other hand, gauges the accuracy of the predicted quantiles or intervals. This targeted approach in using MAE for the mean model, and MPIR and MQL for the upper and lower models, ensures a comprehensive and nuanced assessment, crucial for applications in fields like energy management, conservation strategies, and routine consumption forecasting. By applying these metrics appropriately, the tri-model framework facilitates an in-depth and accurate analysis, enhancing the decision-making process in these essential sectors.

Chapter 4: Evaluation

In the next chapter, we present the practical phase of our research, presenting the experiments conducted to evaluate federated learning and split learning in power prediction. This section outlines our experimental approach, from setup and data handling to the analysis of results, aiming to empirically validate the theoretical insights discussed earlier. Through this hands-on investigation, we seek to illustrate the practical advantages and limitations of each learning methodology, contributing to the broader understanding of machine learning applications in energy management.

4.1 Purpose of the Experiments:

This research aims to conduct a comparative analysis of federated learning and split learning within the context of neural networks for power prediction, a field increasingly dependent on advanced machine learning techniques for enhanced accuracy and efficiency. By exploring the practical aspects and performance metrics of both methodologies in power prediction, the study seeks to precisely evaluate their capabilities in managing large datasets and their effectiveness in forecasting power consumption patterns. The comparison focuses on determining which methodology, federated learning or split learning, provides a more effective framework for power prediction tasks, considering model performance metrics such as prediction accuracy, computational demands, and data privacy management.

Machine learning applications for power prediction are expected to make significant contributions to both theoretical and practical knowledge. This research aims to highlight the strengths and limitations of each approach. The goal is to pave the way for future advancements in developing sophisticated, efficient, and privacy-aware models for power prediction, guiding the evolution of methodologies to meet the sector's demands for high accuracy, data privacy, and computational efficiency.

4.1.1 Rationale for Experimental Focus

The decision to conduct one experiment on federated learning and three on split learning was driven by unique research objectives. Federated learning required only a single, comprehensive experiment to illustrate its decentralized approach and essential performance metrics. Contrarily, split learning necessitated a more thorough investigation due to its novel characteristics and potential for enhancing efficiency and data privacy in power prediction. The choice to vary the neural network's split layer in each of the three split learning experiments is key to understanding how different configurations influence model performance, with a particular focus on accuracy, computational load, and data privacy. This approach aims to uncover the optimal balance between computational efficiency and prediction accuracy, a critical consideration for the practical deployment of power prediction models. Additionally, these experiments collectively seek to provide a detailed insight into how split learning can be

fine-tuned for power prediction, a necessity in the face of the energy sector's increasing data volumes and the urgent need for models that process data efficiently while safeguarding privacy and security.

4.1.2 Analyzing the Impact of Split Layers in Neural Networks for Enhanced Split Learning

Building on the comparative analysis between federated learning and split learning, this research emphasizes the strategic focus on split learning's unique configurations, particularly the variation of the split layer, to optimize neural network architectures for power prediction tasks. This exploration, aligned with assessing the efficiency and effectiveness of these learning paradigms, highlights the importance of nuanced considerations in advancing machine learning within the energy sector. The concept of split layers, marking the division in the network between client-side initial layers and subsequent server-side layers, plays a crucial role in this architecture. It serves as a critical juncture for data processing, where information processed up to this point is sent from the client to the server for further processing. The careful selection and adjustment of the split layer directly contributes to our understanding of how split learning can be tailored to meet computational efficiency and data privacy demands, essential for the practical deployment of predictive models in energy management. This approach not only complements the insights from the federated learning experiment but also provides a comprehensive view of the potential and challenges in implementing advanced machine learning techniques for power prediction, especially in scenarios where balancing computational load and privacy concerns are paramount.

4.1.3 Impact of Different Split Layers on Model Performance

When it comes to power prediction tasks, the placement of the split layer within a neural network plays an important role in determining computational efficiency, data privacy and security, and model accuracy. By studying how the adjustment of the split layer impacts these key performance metrics, we aim to highlight the necessary balance required to optimize models for practical use in power prediction.

Performance in Power Prediction Tasks:

1) Computational Efficiency: The placement of the split layer significantly impacts the computational load, with an earlier split reducing the computation required on the client side—vital for devices with limited processing capabilities. Conversely, a later split shifts the computational burden to the server, increasing demands on the client.

2) Data Privacy and Security: Given the sensitivity of data in power prediction, adjusting the split layer allows for control over the data's granularity and sensitivity being transmitted. An

earlier split results in more raw data being sent to the server, while a later split means that more processed, abstract data is transmitted, potentially enhancing data privacy.

3) Accuracy and Learning Efficacy: The position of the split layer affects the network's learning dynamics. A split too early may not provide the server with sufficient abstracted information, possibly diminishing model accuracy. On the other hand, a later split could improve accuracy but incur higher computational and data transmission costs.

4) Adaptability and Flexibility: Through experimentation with different split layers, the adaptability and flexibility of the split learning model in various power prediction scenarios can be assessed. This flexibility is essential for applications in the real world, where operational conditions and data availability can significantly fluctuate.

Understanding how to effectively deploy machine learning models that meet the diverse needs of real-world applications is crucial, from managing resources efficiently to safeguarding sensitive information.

4.2 Dataset

The datasets (Aupke, 2023) utilized in the federated and split learning experiments contain 129,245 samples, demonstrated in Table 4.2 below, each representing the power consumption data from seven unique Households over a specified period. This data collection is necessary for accurately forecasting power usage.

House Dataset	Number of Samples
Prosumer 1	18,889
Prosumer 2	19,221
Prosumer 3	19,312
Prosumer 4	19,311
Prosumer 5	19,287
Prosumer 6	13,914
Prosumer 7	19,311
Total	129,245

Table 4.2: An illustration of the Dataset used in the experiment

To conduct federated and split learning experiments, the datasets were divided into training and test sets according to best practices in machine learning; that is, 80% of the data from each Prosumer dataset was reserved for model training, while the remaining 20% was set aside for testing. This strategic division ensures that the models are exposed to a significant volume of data during the training phase, enabling them to learn and identify the underlying patterns in power consumption accurately. The test sets, comprising data not encountered by the models during training, provide an unbiased basis for assessing the models' predictive performance and their ability to generalize to new, unseen data. This approach to dataset segmentation supports the comparative analysis between federated learning and split learning, ensuring a fair and realistic evaluation framework. The experiments' outcomes shed light on the most suitable and efficient machine learning frameworks for power prediction, taking into account factors such as prediction accuracy, operational efficiency, and the privacy and security of the data involved.

4.3 Federated Learning Experiment

4.3.1 Experimental Setup

The experimental setup for the federated learning aspect of this study was meticulously designed to explore power prediction across seven distinct Prosumers, each contributing a dataset that spans two years of power usage intertwined with atmospheric conditions. This comprehensive data compilation, which includes detailed power metrics alongside atmospheric variables such as temperature, humidity, and solar radiation, lays a solid foundation for predictive modeling. Utilizing a multi-layered sequential neural network architecture, the experiment was tailored to discern complex patterns in energy consumption effectively. To facilitate a nuanced analysis, three separate models were constructed to forecast the upper, lower, and mean power consumption values, thereby offering an intricate view of Prosumer energy utilization patterns.

The division of datasets into training and testing subsets, with an 80%-20% split, was a strategic choice to ensure model validation on unseen data, a critical step for gauging the models' predictive precision. The training phase leveraged the Adam optimizer, chosen for its efficiency with a learning rate set at 0.01, and employed the Mean Squared Error Loss (MSELoss) function, which is well-suited to the regression-oriented nature of this task, focusing on the accurate prediction of total power consumption. This experimental framework was pivotal in assessing the efficacy of federated learning in capturing and predicting diverse power usage patterns under varying atmospheric influences.

4.3.2 Implementation Details

Python and the PySyft library were used for our federated learning experiment implementation due to their robust support for secure and privacy-focused machine learning. The primary challenge encountered was ensuring effective training across diverse local datasets from each client to enhance the global model while maintaining data privacy and synchronization. The PySyft library facilitated this process by providing the necessary tools to handle the complexities of federated learning, allowing for the integration of varied insights into a cohesive model without compromising on security. This approach was key to navigating the intricacies of decentralized data training and achieving a synchronized, privacy-preserving global model.

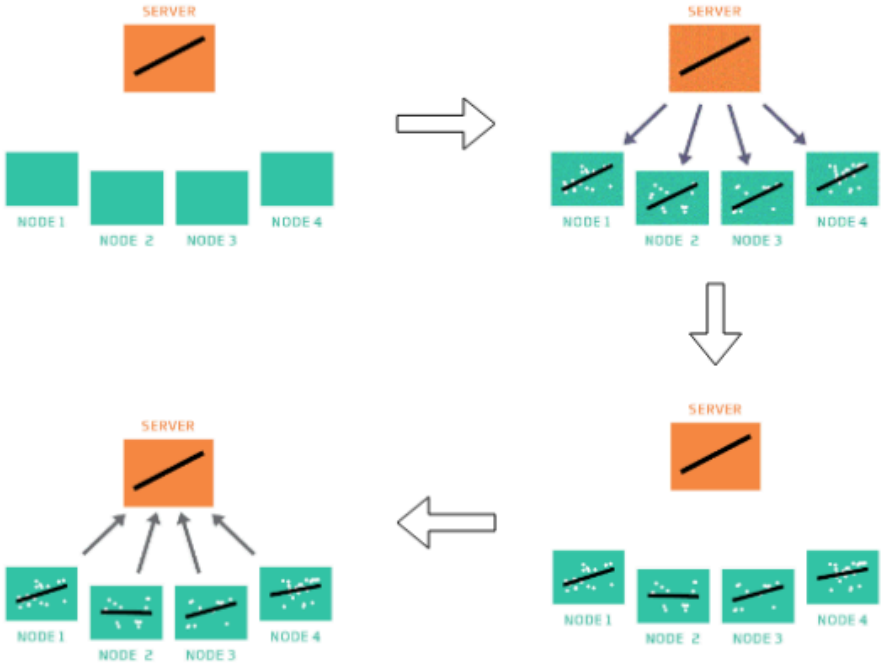


Figure 4.3.2: Federated Learning (Williams, 2018)

In our experiment involving seven clients, each with a distinct Prosumer dataset, the Federated Averaging (FedAvg) algorithm played a pivotal role in training the global model. The process was structured into iterative rounds, encompassing several key steps:

1. **Initialization**: The experiment commenced with the server dispatching the current global model to all seven clients. This initial model acted as the baseline for subsequent local training sessions.
2. **Local Training**: Each client, identified as Node 1 through Node 7 and managing their respective datasets, proceeded to train the model locally. This phase was essential for processing each Prosumer's sensitive power consumption data securely on the client's device, thereby ensuring privacy protection.

3. **Local Updates**: Post-training, clients generated updates reflecting the model's parameter adjustments, informed by their unique data insights. These updates, embodying the local learning, were prepared for transmission without revealing any underlying data.

4. **Uploading Updates**: Subsequently, clients transmitted their model parameter updates back to the server. This phase maintained the federated learning model's integrity by sharing only the parameter updates, not the raw data.

5. **Aggregation**: The server, upon collecting all local updates, proceeded to aggregate these contributions to refine the global model. This aggregation process, typically through averaging the parameter updates, is the essence of the 'Federated Averaging' approach.

6. **Distribution**: Finally, the server redistributes the enhanced global model to the clients, setting the stage for another iteration of training.

This structured approach allowed for the collaborative yet private enhancement of the global model, leveraging the FedAvg algorithm's capabilities to integrate diverse local learnings into a unified predictive model. The iterative process, repeated across multiple rounds, represented a complete cycle from the server distributing the model to receiving the updated parameters, thereby incrementally improving the global model. It learned from various Prosumer power consumption patterns without compromising data privacy. The rounds persisted until the global model achieved a predefined accuracy level or after completing a set number of rounds, culminating in a robust model adept at predicting total power consumption. This model, tested on a 20% dataset holdout, demonstrated its ability to generalize well to new data. Through the FedAvg algorithm, the experiment showcased a scalable machine learning approach, enabling multiple clients to collaboratively train a global model while preserving the confidentiality of their data, an essential feature for sensitive areas like power consumption prediction.

4.4 Split Learning Experiments

4.4.1 Overview

The decision to conduct three distinct split learning experiments was driven by the objective of examining the effects of various neural network split configurations on the efficacy of power consumption prediction models. By altering the split layer, the study aimed to delve into the implications of distributing the computational workload between the client and server on essential factors such as model accuracy, data privacy, and operational efficiency. This exploration is pivotal in understanding the optimal balance required for effective power prediction, ensuring that models are accurate, adhere to privacy standards, and are computationally feasible.

4.4.2 Experiment 1

Setup

In the first split learning experiment, the neural network was segmented into two distinct parts: the initial portion situated on the client side and the subsequent portion on the server side, a configuration applied uniformly across seven clients. Each client utilized datasets from separate Prosumers, aiming to forecast total power consumption. On the client side, there is a single layer responsible for preliminary data processing in the model. The more intricate computational tasks were allocated to the layers on the server side, facilitating the completion of the prediction model's complex computations. This section aims to investigate the impact of distributing computational workload on model performance, specifically in terms of accuracy, efficiency, and privacy.

Implementation

For the implementation phase of the first split learning experiment, preprocessing of each client's dataset was crucial, employing robust scaling techniques to effectively manage outliers. This ensured the data was optimally prepared for model training, with an 80/20 split between training and testing sets to validate model performance on unseen data. The client-side models, each containing the initial processing layer, were developed using PyTorch, a choice that facilitated detailed model customization and efficient computation.

On the server side, the model architecture was designed to receive the intermediate outputs from the client-side models, perform further processing, and generate the final power consumption predictions. The training process was guided by the Adam optimizer, selected for its efficiency, with a learning rate set at 0.01, and the Mean Squared Error Loss (MSELoss) function, aligning with the experiment's focus on predictive accuracy.

A significant challenge encountered during implementation was ensuring seamless communication between the clients and the server, particularly in transferring the intermediate outputs with minimal latency. This was critical for maintaining the experiment's overall efficiency and effectiveness. Moreover, the server's ability to accurately concatenate outputs from all clients before making the final prediction was essential, requiring precise coordination and robust data handling mechanisms to achieve the desired predictive performance.

Expected Outcomes

The anticipated results included assessing the model's prediction accuracy through metrics like the mean absolute error (MAE). The configuration was expected to offer insights into the trade-offs between the computational burden on the clients and the prediction performance.

The advantage of having only the initial layer on the client side in this first experiment was to reduce the client's computational load, making the approach suitable for scenarios with limited client-side resources. Moreover, this setup could potentially enhance data privacy, as only intermediate representations of the data were transmitted to the server, not the raw data itself. This experiment laid the groundwork for analyzing how such a split affects the overall system's efficiency and the accuracy of power consumption predictions.

4.4.3 Experiment 2

Setup

In the second experiment of our split learning series, the setup evolved from the initial configuration by assigning two layers of the neural network to the client side, with the rest of the layers positioned on the server side. This adjustment was uniformly applied to all seven clients, each utilizing a distinct Prosumer dataset to forecast power consumption. The rationale behind allocating two layers to the client side was to investigate a balanced approach between computational demand and model efficacy. This intermediate configuration was designed to assess whether a slightly increased computational load on the client could lead to improved model performance, without significantly taxing the client's resources. It aimed to find an optimal compromise that might be well-suited for scenarios with moderate client-side computational capabilities, exploring how this adjustment affects the overall dynamics of prediction accuracy, efficiency, and data privacy within the split learning framework.

Implementation

In the implementation phase of the second split learning experiment, clients adhered to the established preprocessing routine, employing robust scaling to their datasets and upholding the 80/20 division for training and testing. This experiment expanded the client-side model to include two initial layers dedicated to local data processing, to explore the effects of an increased computational load at the client level. The remaining layers, situated on the server, were tasked with finalizing the predictive analysis.

Following the procedural blueprint of the first experiment, this setup utilized the PyTorch framework to construct the models, capitalizing on the Adam optimizer with a set learning rate of 0.01 and the Mean Squared Error Loss (MSELoss) function to steer the training process. A pivotal aspect of this experiment was the examination of how adding an extra layer to the client-side model influences the overall system, particularly in terms of communication overhead and the complexity of the data being transmitted to the server. This focus aimed to assess the balance between enhancing local processing capabilities and managing the efficiency and effectiveness of data exchange and model performance in the split learning context.

Expected Outcomes

The expected outcomes for this experiment were twofold. First, it aimed to measure any improvements or detriments in prediction accuracy as a result of the additional layer on the client side. Second, it sought to evaluate the balance between data privacy and model performance, hypothesizing that an extra layer could enable the client model to capture more nuanced features locally, thereby reducing the need to transmit potentially sensitive information to the server. The advantages of this two-layer client-side configuration could include improved data representation before transmission and a potential reduction in server-side computational requirements. This experiment was crucial for determining the optimal layer split for scenarios with specific resource allocations between clients and servers, which is a common consideration in distributed energy management systems. In conclusion, Experiment 2 was expected to contribute valuable data to identify the most effective and efficient split learning architecture for power prediction tasks, taking into account both the computational capabilities of clients and the privacy of the data involved.

4.4.4 Experiment 3

Setup

Experiment 3 in our series of split learning trials ventured to further redistribute the computational load, this time significantly towards the client side. This novel configuration placed three layers of the neural network on the client side, leaving only the final layer on the server side. Conducted across the same cohort of seven clients, each utilizing their unique dataset for power consumption prediction, this setup aimed to thoroughly examine the implications of increased client-side computation on the system's overall performance. The intent was to explore how such a configuration affects prediction accuracy, data privacy, and computational efficiency, providing insights into the scalability and practicality of split learning models in environments with varying computational resources.

Implementation

In the implementation phase of Experiment 3, clients adhered to the previously established preprocessing protocol, which included robust scaling and maintaining an 80/20 split between training and testing data. The strategic increase to three layers on the client side was designed to push the boundaries of client computational capacity and assess the enhanced ability of the client-side model to process and extract more complex features directly from the data before any transmission to the server.

Consistent with the methodologies applied in the earlier experiments, the PyTorch framework was utilized for model development, leveraging the Adam optimizer and the Mean

Squared Error Loss (MSELoss) function to guide the training process. A focal point of this experiment was to scrutinize the impact of heightened computational demands on the clients, examining how this redistribution of processing responsibilities influences the overall training dynamics, prediction accuracy, and efficiency of the split learning model. This setup aimed to provide deeper insights into the feasibility and effectiveness of allocating more complex computational tasks to the client side within the split learning architecture.

Expected Outcomes

The primary outcomes of interest for Experiment 3 were to assess whether a more substantial client-side computation would lead to higher accuracy in power consumption predictions and to understand the impact on the server's role in the final prediction phase.

This configuration's advantages could lead to a lowered requirement for data transmission between clients and the server, potentially enhancing data privacy and minimizing network bandwidth consumption. Furthermore, it could empower clients to conduct more localized, in-depth analyses, fully utilizing the unique patterns in their data before forwarding the outcomes to the server for the ultimate prediction. This experiment was vital for assessing the viability of deploying sophisticated models on the client side, especially in situations where protecting data privacy is paramount, and clients possess significant computational resources. The insights gained from this experiment are expected to play a crucial role in identifying the optimal equilibrium between client-side processing and server-side aggregation in power prediction endeavors, offering key understandings of the split learning model's scalability and applicability within energy management frameworks.

Chapter 5: Analysis of Results and Discussion

5.1 Overview of Findings

This chapter provides an in-depth analysis of the results of the federated learning experiment and the three split learning experiments. It aims to evaluate and compare the performance of each model, focusing on its accuracy, computational efficiency, and compliance with data privacy principles. The insights gathered are critical in determining the best learning methods for predicting power consumption in residential environments. Through this examination, we strive to highlight the strengths and limitations of each paradigm to guide future research and practical applications in the field of home energy management.

5.2 Federated Learning Results

The outcomes of the federated learning experiment, which involved data from seven distinct Prosumers, highlighted the need for investigating other learning approaches, particularly split learning. Employing a multi-layered sequential model designed to forecast upper, lower, and mean levels of power consumption, the experiment presented a range of performance results.

These outcomes illuminated the varied efficacy of the model in different consumption scenarios, suggesting that while federated learning holds promise, it may not fully capture the complexity of power usage patterns across diverse Prosumer settings. Such insight prompted a shift in our research focus, leading us to delve into the potential of split learning as a more fitting solution for the intricacies of residential power prediction tasks.

In particular, the Lower Quantile Model's loss of 45.5770 on the test data highlighted difficulties in accurately predicting lower ranges of power usage—a crucial aspect for energy-saving strategies and demand-response initiatives. Accurate predictions in this range are vital for optimizing energy consumption and reducing waste. The Upper Quantile Model demonstrated somewhat improved performance, with a loss of 30.1327, suggesting a relatively more accurate forecast of peak power usage. Despite this improvement, the need for further refinement to enhance peak consumption forecasting accuracy was apparent.

The Mean Quantile Loss (MQL) for the combined model was recorded at 37.8548, alongside a mean absolute error (MAE) of 87. While these metrics reflect the model's potential to a certain degree, they did not meet the accuracy and efficiency benchmarks essential for dependable power prediction. These figures pointed to a model that, despite leveraging the privacy-preserving and decentralized attributes of federated learning, struggled to comprehensively understand the nuances of varied Prosumer energy consumption patterns.

Given these outcomes, we found that although federated learning has its advantages in maintaining data privacy and managing decentralized datasets, it did not meet our expectations in terms of model performance and precision. This realization prompted a shift towards investigating split learning in subsequent experiments. Split learning, characterized by its unique method of dividing neural network processing between the client and server, promised a more customized solution to the computational and accuracy challenges observed in the federated learning experiment. This strategic pivot aimed to explore methods that could more adeptly balance computational efficiency, data privacy, and predictive accuracy in power consumption forecasting.

5.3 Evaluation of Split Learning Performance

5.3.1 Experiment 1

In the initial split learning experiment, the outcomes observed across the seven clients showcased the model's nuanced performance, specifically designed to forecast power consumption. This experiment employed a neural network architecture where one layer was situated on the client's side, and the remaining layers were hosted on the server side. This partitioning facilitated a diverse array of results that warrant detailed scrutiny. The arrangement allowed for an exploration of how varying the computational load between client and server

impacts the model's effectiveness in different Prosumer scenarios, highlighting the adaptability and potential challenges of implementing split learning for energy management tasks. The presented table shows different outcomes that require a thorough examination to understand the model's performance nuances.

	Client 1	Client 2	Client 3	Client 4	Client 5	Client 6	Client 7
Mean Quantile Loss for Upper Quantile (75th percentile)	6.93	3.48	5.47	2.6	14.38	5.93	3.75
Mean Quantile Loss for Lower Quantile (25th percentile)	18.15	8.84	8.42	3.77	22.4	4.2	3.95
Combined Mean Quantile Loss	12.54	6.16	6.95	3.19	18.39	5.07	3.85
Mean Prediction Interval Range (MPIR)	96.33	48.15	47.6	21.79	124.99	39.46	27.04
MAE	12.03	8.41	12.42	3.49	15.08	4.68	6.48

Table 5.3.1: Results of Experiment 1

Key Takeaways

Mean Quantile Loss (MQL): The MQL for both upper and lower quantiles showcases significant variability across different clients, underscoring the model's sensitivity to the unique data characteristics of each Prosumer. For example, Client 1 exhibits a higher MQL for the upper quantile than Client 2, suggesting variances in the predictability of peak power usage patterns among different Prosumers.

Mean Prediction Interval Range (MPIR): MPIR values exhibit considerable variation among clients, with Client 5 displaying an especially wide range. A broad MPIR indicates a higher degree of uncertainty in the predictions, which may pose challenges for applications necessitating accurate power management.

Mean Absolute Error (MAE): The MAE values reveal that the model achieves high accuracy for certain clients (e.g., Client 4 with an MAE of 3.49) but is less accurate for others (e.g., Client 5 with an MAE of 15.08). With an overall mean MAE of 8.94 across all clients, the model generally performs well, though there is noticeable variability in accuracy among individual clients.

Interpretation of Result

The results of the split learning experiments highlight the model's adeptness in handling the complex dynamics of Prosumer power prediction scenarios, revealing a distinct difference in power consumption patterns among various Prosumers. Based on the data in the table, our analysis provides a comprehensive understanding of the split learning model's ability to accurately forecast power consumption in diverse environments. Each client showcased unique characteristics in their power usage, highlighting the model's capacity to capture these individual patterns significantly. The variation in Mean Quantile Loss (MQL) for both upper and lower quantiles, alongside the Mean Absolute Error (MAE), reflects the model's adaptability to heterogeneous data.

By design, the split learning approach facilitates modeling by partitioning neural network processing between the client and the server. This partitioning effectively localizes the learning process, enabling the model to tailor its predictions to the specific power usage behaviors of each Prosumer. Such customization is crucial for achieving accurate power prediction tasks, as it moves beyond a one-size-fits-all solution to address the complexity of residential power consumption directly.

The observed variation in MQL and MAE across clients not only underscores the model's ability to navigate the complexity of residential power consumption but also demonstrates its robustness in understanding and predicting distinct energy usage profiles. This adaptability is

essential for models tasked with accurate power prediction, making split learning a potent tool for managing the nuanced requirements of Prosumer power prediction tasks.

Comparison to Federated Learning

In comparing the outcomes of Split Learning Experiment 1 with Federated Learning, it becomes evident that split learning significantly outperforms federated learning in terms of prediction accuracy for residential power consumption. The split learning model, which requires the client to train only one layer of the neural network, achieved a mean absolute error (MAE) of approximately 8.94 across all clients. This contrasts sharply with the federated learning model, where clients are tasked with training the entire model, resulting in a much higher MAE of 87. This stark difference in MAE underscores the superior accuracy of the split learning approach, making it a more viable option for Prosumers equipped with devices possessing limited computational power. Given that federated learning fell short of achieving the desired benchmarks for accuracy and efficiency, the split learning model emerges as a more effective solution for power prediction tasks in Prosumer scenarios. It offers a promising balance between computational efficiency, data privacy, and predictive accuracy, highlighting its potential as an advantageous approach for environments with constrained computational resources.

Visual Representation

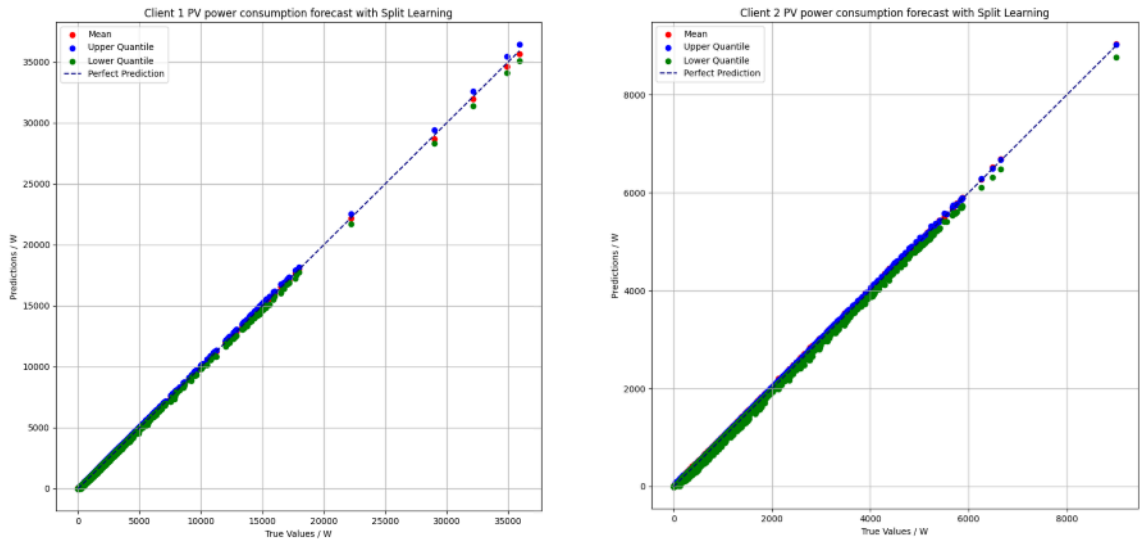


Figure 5.3.1a: Scatter Plot of Clients 1 & 2

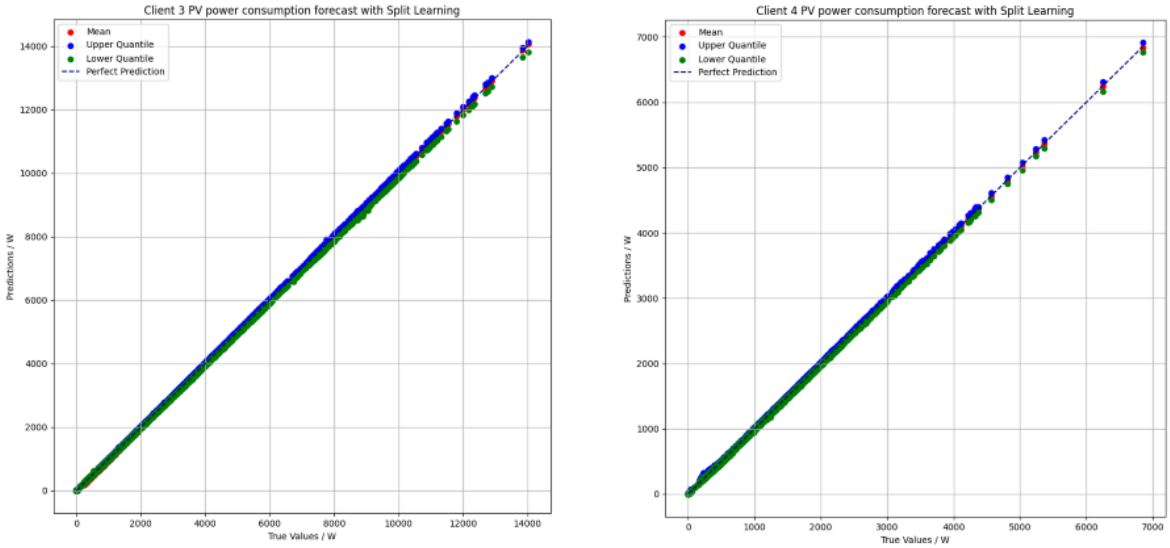


Figure 5.3.1b: Scatter Plot of Clients 3 & 4

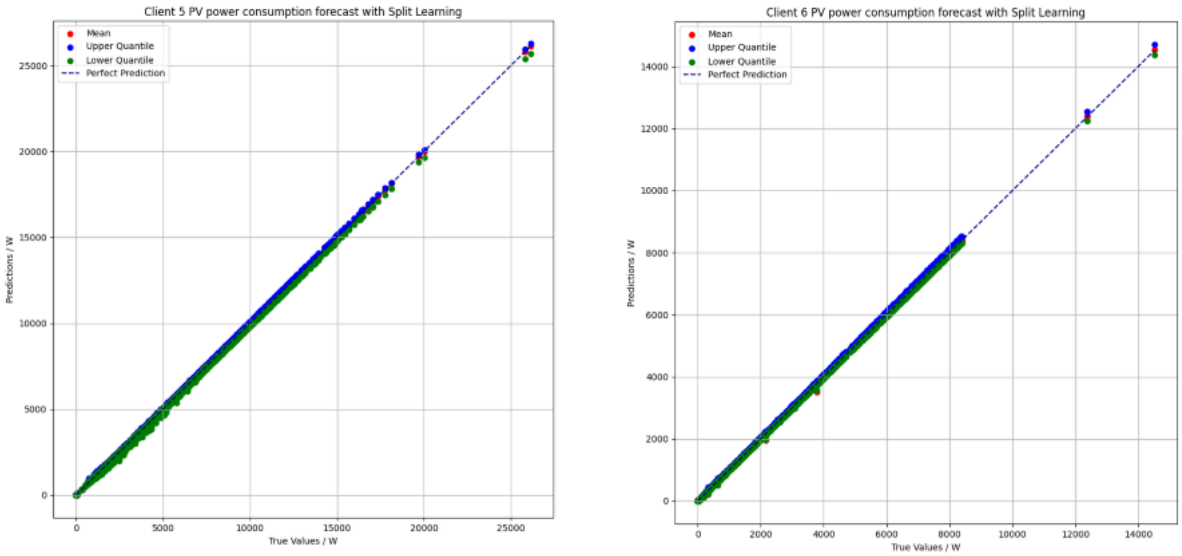


Figure 5.3.1c: Scatter Plot of Clients 5 & 6

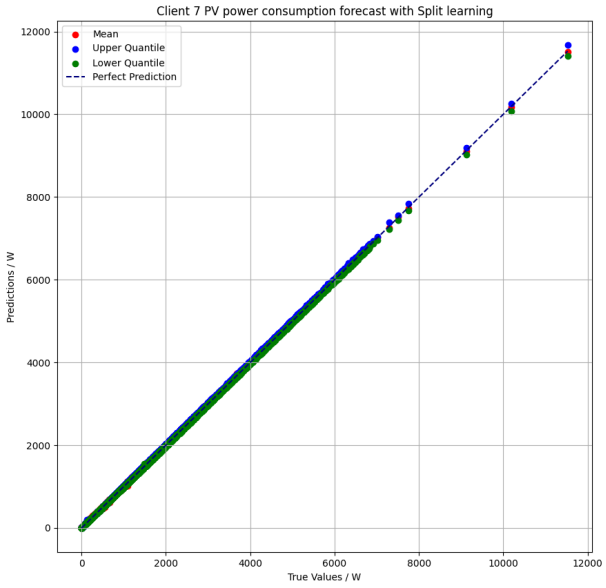


Figure 5.3.1d: Scatter Plot of Client 7

The scatter plots for Clients 1 through 7 represent the predictive performance of our split learning model against the test data, which constitutes 20% of each client's dataset. This test data is critical as it evaluates the model's ability to generalize to new data after the training phase.

In summary, the scatter plot analysis for seven clients in our split learning experiment succinctly demonstrates the model's forecasting accuracy across different power consumption scenarios, especially at higher consumption levels. These plots validate the model's ability to generalize to new data, showcasing its effectiveness in capturing unique Prosumer energy patterns. The analysis emphasizes the importance of tailoring models to specific Prosumer needs for optimal performance. This evaluation confirms the model's suitability for real-world energy management and conservation, underlining its potential to enhance energy efficiency efforts significantly.

5.3.2 Experiment 2

In Experiment 2, where two layers of the neural network were allocated to the client side, the results offer insightful data on split learning's performance in power consumption prediction across seven clients. The mean absolute error (MAE) across all clients in this setup is approximately 9.57, marking a slight increase from the mean MAE of 8.94 observed in Experiment 1, which had only one layer on the client side.

	Client 1	Client 2	Client 3	Client 4	Client 5	Client 6	Client 7
Mean Quantile Loss for Upper Quantile(75th percentile)	8.75	8.03	7.74	5.06	18.15	3.59	7.12
Mean Quantile Loss for Lower Quantile(25th percentile)	7.74	3.71	5.57	4.09	15.38	2.57	4.62
Combined Mean Quantile Loss	8.248	5.87	6.66	4.58	16.76	3.08	5.87
Mean Prediction Interval Range (MPIR)	19.58	28.86	27.66	35.44	63.28	14.04	23.22
MAE	20.77	5.04	8.95	4.52	15.43	5.29	6.96

Table 5.3.2: Results of Experiment 2

Key Takeaways

The Mean Quantile Loss (MQL): Reflects variability for both upper and lower quantiles, similar to Experiment 1, highlighting the model's performance dependency on the unique data of each client.

The Combined MQL: Shows variation among clients, with Client 6 exhibiting improved performance, contrasting with Client 5, which demonstrates less accuracy.

Mean Prediction Interval Range (MPIR): Indicates significant differences across clients, suggesting variability in prediction confidence.

Mean Absolute Error (MAE): Ranges widely among clients, with Client 1 notably experiencing a higher MAE in this experiment, pointing to the impact of allocating two neural network layers to the client side on prediction accuracy across Prosumers.

Interpretation of Result

The increased MAE in Experiment 2 could indicate the model's complexity affecting its generalization capability. This suggests that simply allocating additional layers to the client side does not guarantee improved performance and might introduce new challenges. The variability in Mean Prediction Interval Range (MPIR) across clients in Experiment 2 underscores that the model's prediction confidence can significantly vary based on the specific client's dataset. Consequently, Experiment 2 demonstrates that localizing more layers on the client side does not uniformly enhance predictive performance. While some clients may benefit from this configuration, it could result in diminished accuracy for others, as reflected by the overall rise in Mean Absolute Error (MAE). This observation implies a need for a carefully balanced distribution of neural network layers between the client and server, optimized individually for each client to attain optimal outcomes. The split learning model's sensitivity to the distinct energy usage patterns of each Prosumer is evident, emphasizing the critical need for customizing the model's complexity according to the dataset at hand to effectively capture and predict power consumption patterns.

Comparison to Experiment 1

Comparing Experiment 2 to Experiment 1 reveals that the overall mean MAE has risen, indicating that adding a layer to the client side did not consistently enhance model performance across all clients. Additionally, the performance across quantiles appears more balanced in Experiment 2, with the MQL for lower and upper quantiles showing closer values in some instances. This contrasts with Experiment 1, where lower quantiles typically experienced higher losses, suggesting a nuanced shift in model efficacy when adjusting the neural network's client-side complexity.

5.3.3 Experiment 3

In Experiment 3, we delved deeper into the split learning architecture by shifting an additional layer from the server side to the client side, leading to a configuration where three layers were processed at the client level. This modification was designed to investigate how increased complexity on the client side affects the model's precision in forecasting residential power consumption. By reallocating one more neural network layer to the client side from the server, as compared to Experiment 2, the experiment maintained the same overall architecture in terms of layer count but altered the distribution between client and server, resulting in the server processing fewer layers.

	Client 1	Client 2	Client 3	Client 4	Client 5	Client 6	Client 7
Mean Quantile Loss for Upper Quantile (75th percentile)	9.46	4.67	8.55	3.72	15.05	3.78	5.49
Mean Quantile Loss for Lower Quantile (25th percentile)	8.28	2.48	4.78	3.66	36.81	2.05	4.68
Combined Mean Quantile Loss	8.87	3.58	6.67	3.69	25.93	2.91	5.08
Mean Prediction Interval Range (MPIR)	57.98	25.42	36.54	28.44	23.19	4.37	37.11
MAE	30.88	12.73	16.6	11.63	33.21	6.63	6.98

Table 5.3.3: Results of Experiment 3

Interpretation of Result

The outcomes of Experiment 3 underscore a notable trend: as additional layers were transitioned from the server to the client side, the Mean Absolute Error (MAE) across all clients escalated to approximately 16.95, a significant increase from the mean MAE of 8.94 in Experiment 1 (with one layer on the client side) and 9.57 in Experiment 2 (with two layers on the client side). Similarly, the Mean Quantile Loss (MQL) for both upper and lower quantiles, along with the Combined MQL, exhibited variability, with some clients witnessing a marked rise in these metrics. The Mean Prediction Interval Range (MPIR) also showed extensive variation among clients, reflecting diverse levels of prediction confidence.

This uptick in MAE and the variability in MQL and MPIR imply that augmenting the client-side complexity, while simultaneously simplifying the server side, does not invariably improve the model's predictive accuracy. Instead, it seems to compromise the model's generalization ability across various Prosumers. A plausible reason for this pattern is that allocating more layers to the client side leaves the server with less data to analyze, thereby reducing its ability to discern and learn from more complex data patterns. This reduction in server-side processing capability may limit the model's overall ability to generalize, considering the server's integral role in synthesizing insights from multiple clients for broader predictions.

Furthermore, the heightened complexity on the client side might pose computational challenges or inefficiencies, particularly in accurately capturing the data variability within individual Prosumers, especially on client devices with constrained computational capacities. These findings suggest that a more equitable distribution of model complexity, potentially with a bias towards the server side, could be more conducive for tasks that demand generalization across varied datasets.

5.3.4 Conclusions

The three split learning experiments collectively highlight the nuanced balance required in distributing computational complexity between client and server sides for optimal predictive performance in residential power consumption forecasting. Experiment 1 established a baseline, demonstrating the model's capability with minimal client-side complexity. Experiment 2 explored the effects of slightly increased complexity, showing a modest rise in Mean Absolute Error (MAE) but suggesting the potential for more balanced quantile predictions. Experiment 3, with further increased client-side complexity, revealed a significant increase in MAE, indicating that additional complexity does not necessarily translate to improved performance and may hinder the model's generalization ability. Across the experiments, the variability in Mean Quantile Loss (MQL) and Mean Prediction Interval Range (MPIR) underscored the model's sensitivity to individual Prosumer data characteristics. These findings suggest that while leveraging client-side computation can enhance privacy and utilize local data features, there is a critical threshold beyond which additional complexity may detract from the model's effectiveness. The insights from these experiments emphasize the importance of carefully calibrating the split learning model's architecture to strike the right balance between computational efficiency, data privacy, and predictive accuracy, paving the way for tailored and effective energy management solutions in diverse residential settings.

5.4 Evaluation of Predictive Performance Using Cumulative Distribution Functions

Cumulative Distribution Functions (CDFs) provide a detailed perspective on model performance by showcasing the entire range of prediction errors. In this section, we compare the predictive capabilities of a Split Learning model against a traditional Neural Network (NN) model, both customized for individual client datasets in power consumption forecasting. Utilizing CDFs allows us to evaluate not only the average performance but also the full error distribution, offering a complete view of the accuracy of each model. The evaluation extends across multiple clients, each with distinct characteristics and consumption patterns. Through CDFs, we capture the percentage of predictions that fall within specific error thresholds, enabling us to visualize and quantify the probability of achieving a certain level of prediction accuracy across different client scenarios. This client-specific analysis through CDFs provides a holistic understanding of each model's accuracy, highlighting the nuanced performance across the spectrum of prediction errors.

5.4.1 CDF Interpretation

In the interpretation of Cumulative Distribution Functions (CDF) for each client, we analyze two distinct CDF curves (as shown in the figures below): one representing the Split Learning model and the other for the traditional Neural Network (NN) model. The shape and trajectory of these curves provide critical insights into model performance. A curve that rises sharply towards a cumulative probability of 1 (100%) suggests a greater proportion of predictions are made with lower errors, indicating a model's superior performance. On the other hand, a curve that aligns more closely with the diagonal suggests a higher probability of larger errors, denoting a less accurate model. This comparative analysis of CDF curves allows us to assess the precision of the Split Learning and NN models in forecasting power consumption across various clients, highlighting the differences in their ability to minimize prediction errors.

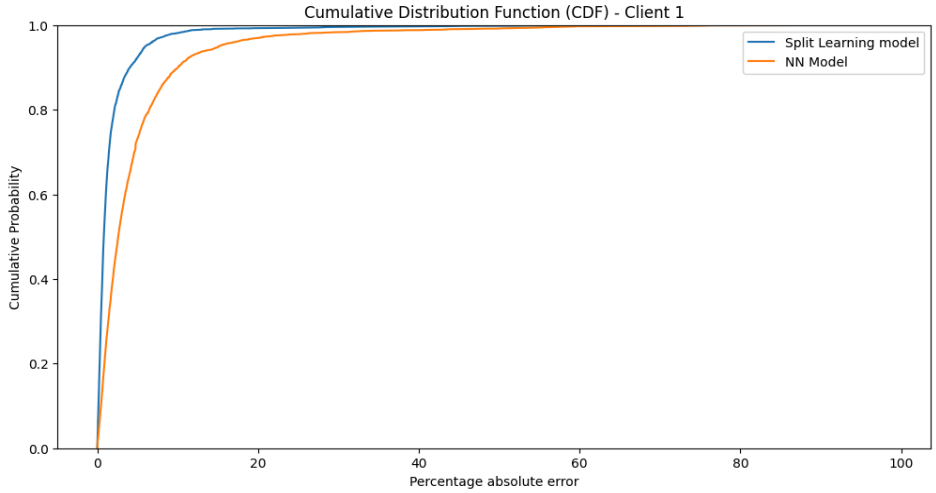


Figure 5.4.1 a: CDF curves for Client 1

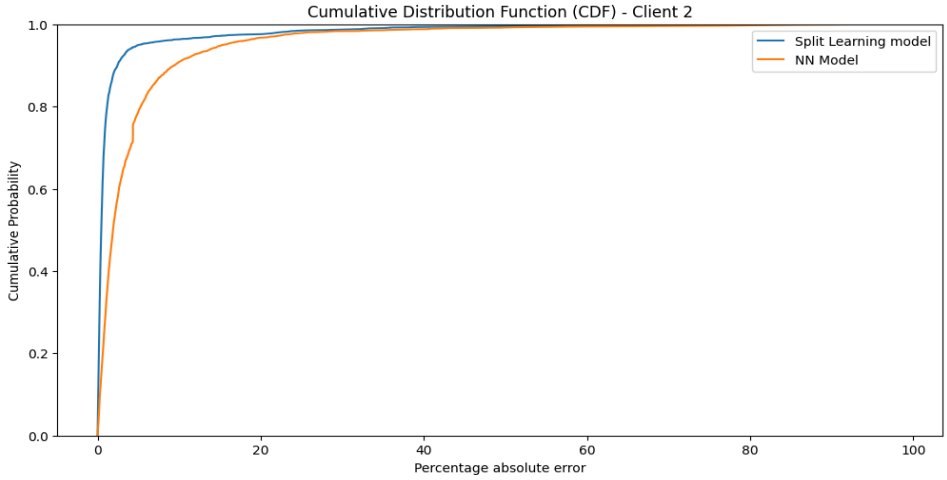


Figure 5.4.1 b: CDF curves for Client 2

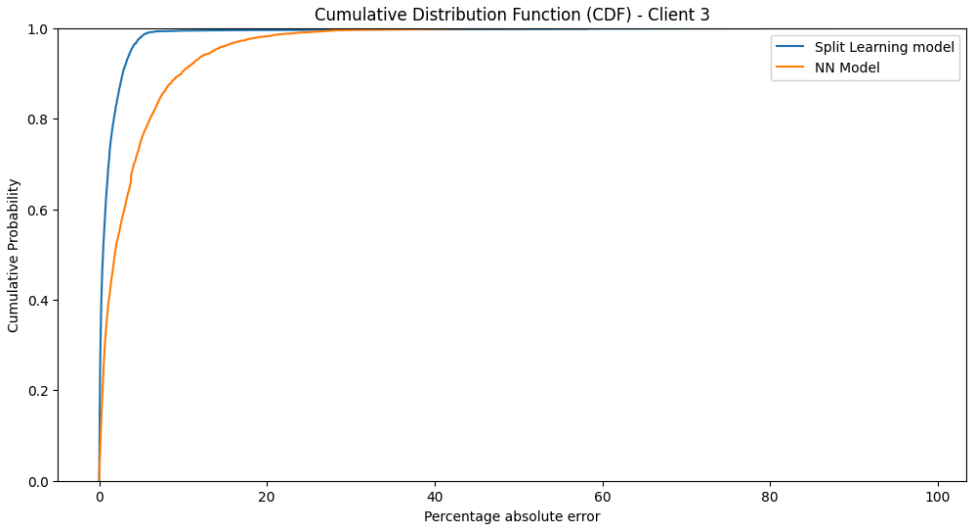


Figure 5.4.1 c: CDF curves for Client 3

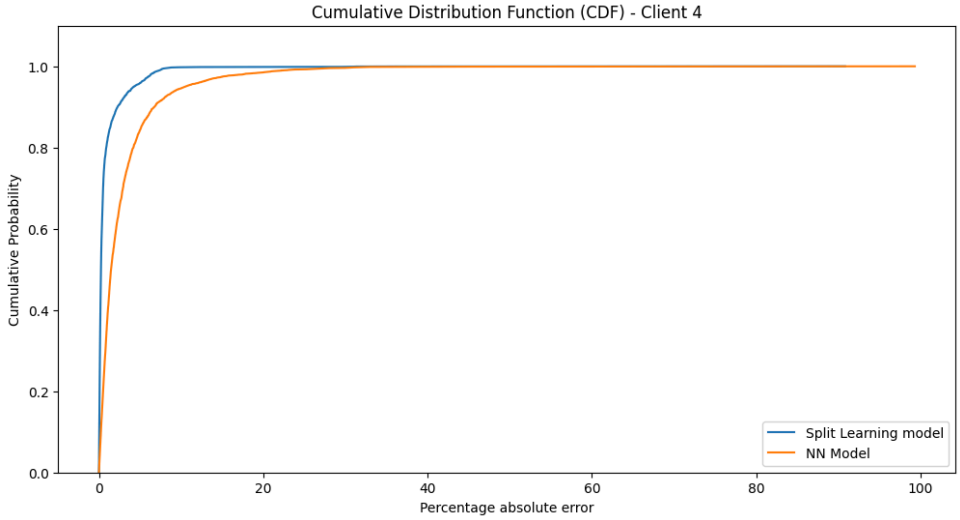


Figure 5.4.1 d: CDF curves for Client 4

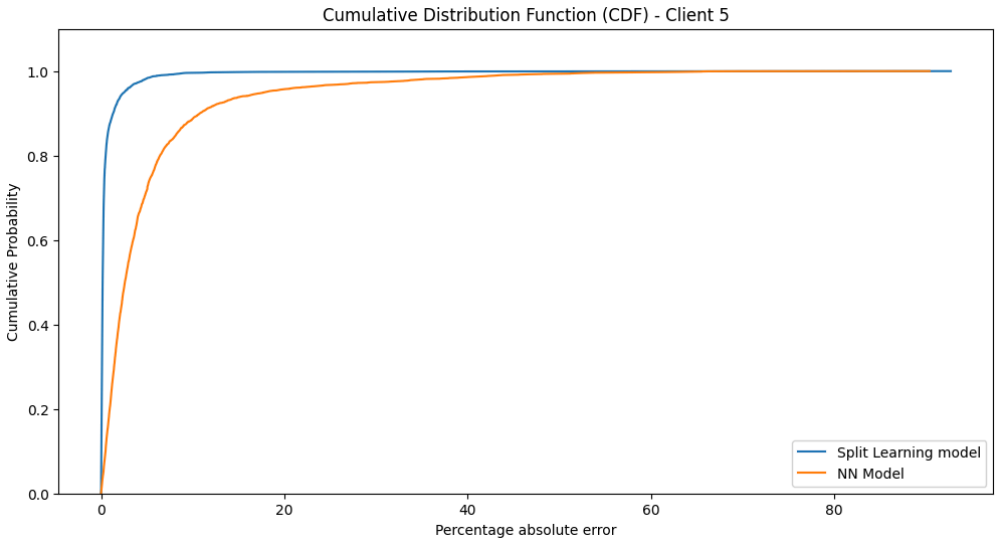


Figure 5.4.1 d: CDF curves for Client 5

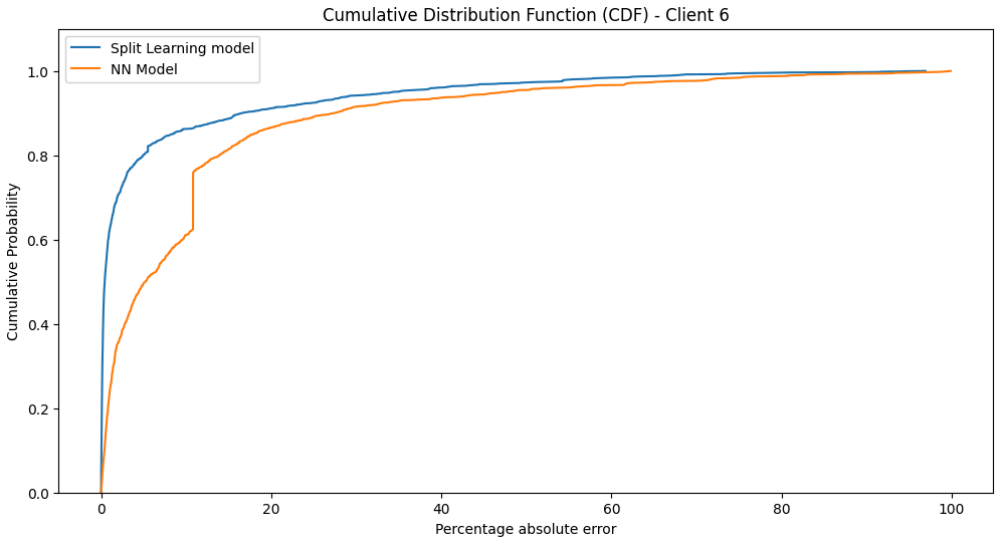


Figure 5.4.1 e: CDF curves for Client 6

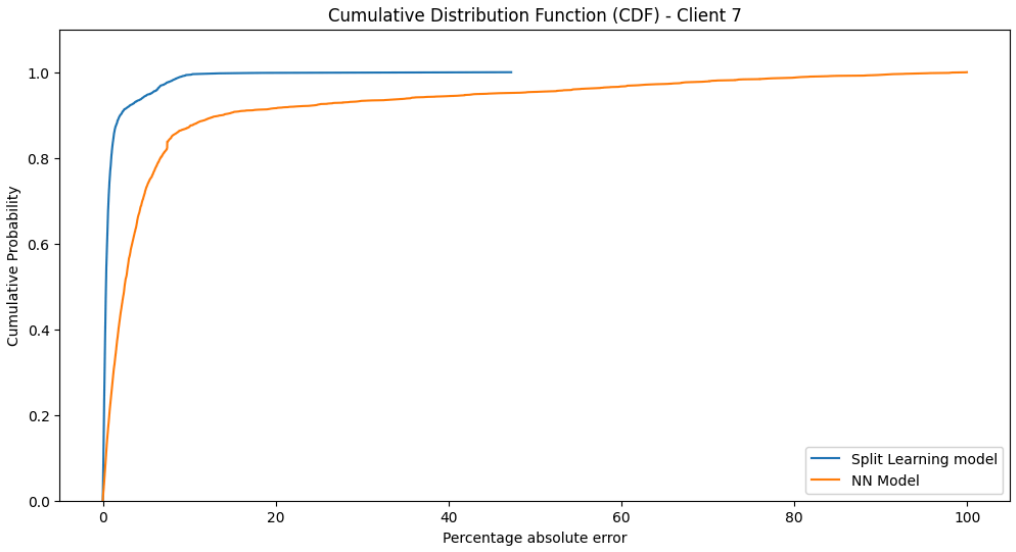


Figure 5.4.1 f: CDF curves for Client 7

5.4.2 Comparative Insights

Across the client datasets, we observe varying degrees of performance differentiation between the Split Learning and NN models. In certain cases, the Split Learning model demonstrates a clear advantage, rapidly approaching a cumulative probability of 1 at lower error percentages. In other instances, the performance of both models converges, suggesting a more comparable predictive capability. The steepness and the shape of the curves provide insights into the consistency and reliability of each model. For example, a CDF that shows a rapid increase and a plateau at a high cumulative probability suggests that the model is consistently accurate, with a high percentage of predictions falling within a tight error margin.

Chapter 6: Conclusion and Synthesis of Experiments

Through our comprehensive investigation that involved federated learning and a series of split learning experiments, we have thoroughly explored the complexities of predictive models in the domain of Prosumer power consumption. Our journey has revealed crucial insights that not only define the current state of predictive modeling but also open up possibilities for future research and practical applications.

The federated learning approach, known for its privacy protection and decentralized nature, has emerged as a new competitor. However, it has shown limitations in capturing the complex energy usage patterns typical of diverse Prosumer environments. Despite its significant privacy advantages, federated learning alone seems inadequate for the intricate task of forecasting power consumption.

Due to the federated learning approach not yielding the results we were hoping for, we turned our attention to split learning experiments. These experiments inspired subtle performances across various client-specific datasets. Experiment 1 established a strong foundation, with the Split Learning model achieving an impressive mean Mean Absolute Error (MAE) of 8.94, demonstrating its capability for precise power consumption modeling. However, as we progressed with Experiments 2 and 3, which incrementally increased the model's complexity on the client side, a more detailed picture of model efficacy emerged. Experiment 2 observed a slight increase in MAE, but Experiment 3 experienced a more significant jump to an MAE of 16.95. This highlighted concerns regarding the drawbacks of adding too much complexity on the client side, which could potentially impair the model's ability to generalize. A notable advantage of split learning emphasized here, is its dual benefit of maintaining data privacy while reducing the computational load on the client side. This is particularly crucial in scenarios with limited client-side resources or where data privacy is paramount. By shifting a significant portion of the computational workload to the server side, split learning models offer a sustainable and privacy-conscious alternative for energy prediction tasks.

When directly comparing the federated learning and split learning outcomes, our analysis reveals a distinct difference in their effectiveness for forecasting Prosumer power consumption. Federated learning faced difficulties in accurately modeling the varied energy consumption patterns across Prosumers, as indicated by a higher Mean Absolute Error (MAE) of 87. On the other hand, split learning showed a significant improvement in predictive accuracy, starting with Experiment 1, which achieved a considerably lower mean MAE of 8.94. This difference became even more pronounced in later experiments, where, despite increasing computational complexity on the client side, split learning continued to outperform federated learning, reaching an MAE of 16.95 in Experiment 3. This comparison not only highlights split learning's superior adaptability and predictive capability across diverse datasets but also points to its potential as a more effective and privacy-aware modeling approach in the energy sector. Additionally, the Mean Quantile Loss (MQL) in the split learning experiments also surpassed that of the federated learning approach, further affirming split learning's improved predictive accuracy at various levels of power consumption.

The conclusion derived from these experimental findings underscores several crucial aspects. Firstly, managing computational complexity in split learning models necessitates a carefully calibrated balance, specifically adjusted to the unique requirements of each scenario to enhance performance. Secondly, the split learning method, particularly when achieving a balanced distribution of layers, proves to be more effective in capturing the diverse consumption patterns observed in residential environments than federated learning. Lastly, the data emphasizes the importance of customizing models to the specific power consumption behaviors of individual Prosumers to achieve the best results. In conclusion, while split learning offers a promising avenue for predictive modeling in residential power consumption, finding the optimal balance between accuracy, computational efficiency, and data privacy continues to be a critical challenge. The insights from these experiments not only affirm the potential of current models

but also pave the way for future advancements in creating intelligent, privacy-conscious, and efficient predictive models for the energy sector.

6.1 Section: Future Work

As we project forward from the conclusions drawn in this study, one area of significant interest for future research is the exploration of model drift in the context of Prosumer power consumption forecasting. Model drift, or the phenomenon where the model's predictive performance degrades over time due to changes in the underlying data distribution, presents a critical challenge in ensuring the long-term accuracy and reliability of predictive models.

6.1.1 Studying Model Drift in Client Data

The dynamic nature of Prosumer energy consumption, influenced by factors such as seasonal changes, evolving consumer behaviors, and alterations in Prosumer composition, necessitates a robust mechanism to detect and adapt to model drift. In the pursuit of maintaining model efficacy, understanding the nuances of how and when these shifts occur becomes paramount.

6.1.2 Advantage of Split Learning in Detecting Model Drift

Within this context, the split learning framework offers a promising avenue for more effectively detecting model drift at the client level. Unlike federated learning, where model updates are aggregated from multiple clients without direct access to individual data patterns, split learning allows for a more granular observation of changes in data at the source. This granularity provides a pivotal advantage: the ability to identify and respond to drift in a timely and client-specific manner, ensuring that the model remains attuned to the evolving consumption patterns of each Prosumer.

6.1.3 Proposed Methodology

Future work will focus on developing methodologies within the split learning framework to systematically detect model drift. This will involve:

- Continuous Monitoring: Implementing mechanisms to continuously monitor prediction accuracy and data distributions for signs of drift at the client level.
- Adaptive Learning: Designing adaptive learning algorithms that can automatically adjust the model in response to detected drift, ensuring sustained accuracy over time.
- Client-Specific Adjustments: Exploring strategies for making client-specific model adjustments, thereby personalizing the model's response to drift based on the unique characteristics of each Prosumer's energy consumption.

6.1.4 Implications and Goals

The ability to effectively manage model drift not only extends the lifespan of predictive models but also enhances their practical utility in real-world applications. By ensuring that models remain reflective of current data trends, we can improve decision-making processes for energy management, optimize energy consumption, and contribute to overall energy efficiency. The ultimate goal of this future work is to solidify the foundation of split learning as a resilient, adaptable, and privacy-preserving methodology for predictive modeling in the energy sector and beyond.

This focus on model drift within the split learning framework promises to unlock new potentials for predictive accuracy, model longevity, and the personalization of energy forecasting models, marking a critical step forward in our ongoing quest to harness the power of predictive analytics in energy management.

References

Abbas, Z., Ivarsson, J. R., Al-Shishtawy, A., & Vlassov, V. (2019). Scaling Deep Learning Models for Large Spatial Time-Series Forecasting. IEEE. https://dx.doi.org/10.1109/BigData47090.2019.9005475

Achuthan, G., & Maksood, F. Z. (2017). Sustainability in Oman: Energy Consumption Forecasting using R. Indian Journal of Science and Technology. https://dx.doi.org/10.17485/IJST/2017/V10I10/97008

Bawdekar, A. A., Prusty, B., & Bingi, K. (2022). Sensitivity Analysis of Stationarity Tests' Outcome to Time Series Facets and Test Parameters. Mathematical Problems in Engineering, 2022, Article ID 2402989. https://dx.doi.org/10.1155/2022/2402989

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). Time Series Analysis: Forecasting and Control (5th ed.). John Wiley & Sons Inc. https://doi.org/10.1111/jtsa.12194

Chang, Y.-W., Chen, H.-Y., Han, C., Morikawa, T., Takahashi, T., & Lin, T. (2023). FINISH: Efficient and Scalable NMF-Based Federated Learning for Detecting Malware Activities. IEEE Transactions on Emerging Topics in Computing. https://dx.doi.org/10.1109/TETC.2023.3292924

Chen, S., Long, G., Shen, T., & Jiang, J. (2023). Prompt Federated Learning for Weather Forecasting: Toward Foundation Models on Meteorological Data. arXiv. https://dx.doi.org/10.48550/arXiv.2301.09152

Dozie, K., & Ibebuogu, C. C. (2023). Decomposition with the Additive Model Using Buys-Ballot Technique of Quadratic Trend-Cycle Component in Descriptive Time Series Analysis. Asian Journal of Probability and Statistics. https://dx.doi.org/10.9734/ajpas/2023/v25i3564

Eliasy, A., & Przychodzen, J. (2020). The role of AI in capital structure to enhance corporate funding strategies. Array, 6, 100017. https://doi.org/10.1016/j.array.2020.100017

Ezzeddine, F., Ayoub, O., Andreoletti, D., Tornatore, M., & Giordano, S. (2023). Vertical Split Learning-Based Identification and Explainable Deep Learning-Based Localization of Failures in Multi-Domain NFV Systems. IEEE. https://dx.doi.org/10.1109/NFV-SDN59219.2023.10329604

Graser, A., Jalali, A. N., Lampert, J., Weissenfeld, A., & Janowicz, K. (2023). Deep Learning From Trajectory Data: a Review of Deep Neural Networks and the Trajectory Data Representations to Train Them. EDBT. https://ceur-ws.org/Vol-3379/BMDA_2023_paper_7556.pdf

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2015). LSTM: A search space odyssey. IEEE Transactions on Neural Networks and Learning Systems, 27(10). https://doi.org/10.1109/TNNLS.2016.2582924

Gusev, A., Chervyakov, A., Alexeenko, A., & Nikulchev, E. (2023). Particle Swarm Training of a Neural Network for the Lower Upper Bound Estimation of the Prediction Intervals of Time Series. Mathematics. https://dx.doi.org/10.3390/math11204342

Hansda, R., & Murmu, R. (2023). Wind Speed Forecasting using Artificial Neural Networks: A Comparative Study. IEEE. https://dx.doi.org/10.1109/ICSCNA58489.2023.10370186

Her, O. Y., Mahmud, M. S. A., Abidin, M., Ayop, R., & Buyamin, S. (2022). Artificial neural network-based short-term electrical load forecasting. International Journal of Power Electronics and Drive Systems. https://dx.doi.org/10.11591/ijpeds.v13.i1.pp586-593

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9 (8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

Huang, X., Ding, Y., Jiang, Z., Qi, S., Wang, X., & Liao, Q. (2020). DP-FL: a novel differentially private federated learning framework for the unbalanced data. World Wide Web, 23(N/A),N/A.10.1007/s11280-020-00780-4https://www.researchgate.net/publication/341059324_DP-FL_a_novel_differentially_private_federated_learning_framework_for_the_unbalanced_data/citations)

Huang, K., et al. (2022). A Privacy-preserving Users' Power Load Prediction Method Based on Federated Learning. In 2022 5th International Conference on Computing and Big Data (ICCBD) (pp. 94-101). Shanghai, China. https://doi.org/10.1109/ICCBD56965.2022.10080551

Hang, F., Xie, L., Zhang, Z., Guo, W., & Li, H. (2022). Research on Privacy Protection Based on Joint Learning in Power Industry Big Data Analysis. Data Generation, Analysis and Encryption Journal. https://dx.doi.org/10.13052/dgaej2156-3306.3759

Hyndman, R. J., & Athanasopoulos, G. (2018). Forecasting: Principles and Practice. OTexts. https://otexts.com/fpp3/

Jiang, L., Wang, Y., Zheng, W., Jin, C., Li, Z., & Teo, S. G. (2022). LSTMSPLIT: Effective SPLIT Learning based LSTM on Sequential Time-Series Data. Retrieved from https://arxiv.org/abs/2203.043055

Kale, S., Sekhari, A., & Sridharan, K. (2021). SGD: The Role of Implicit Regularization, Batch-size and Multiple-epochs. arXiv. https://arxiv.org/abs/2107.05074

Klemm, J., Gabriel, A., & Sill Torres, F. (2023). Predicting Future Wave Heights by Using Long Short-TermMemory.IEEE.https://dx.doi.org/10.1109/OCEANSLimerick52467.2023.10244329

Krishnan, S., Magalingam, P., & Ibrahim, R. (2021). Hybrid deep learning model using recurrent neural network and gated recurrent unit for heart disease prediction. International Journal of Electrical and Computer Engineering. https://dx.doi.org/10.11591/IJECE.V11I6.PP5467-5476

Kolsi, L., Al-Dahidi, S., Kamel, S., Aich, W., Boubaker, S., & Ben Khedher, N. (2023). Prediction of Solar Energy Yield Based on Artificial Intelligence Techniques for the Ha'il Region, Saudi Arabia. Sustainability, 15(1), 774. https://doi.org/10.3390/su15010774

Kumar, D., Mathur, H. D., Bhanot, S., & Bansal, R. (2020). Forecasting of solar and wind power using LSTM RNN for load frequency control in isolated microgrid. International Journal of Electrical Power & Energy Systems, 117, 105623. https://dx.doi.org/10.1080/02286203.2020.1767840

Lazcano, A., Herrera, P. J., & Monge, M. (2023). A Combined Model Based on Recurrent Neural Networks and Graph Convolutional Networks for Financial Time Series Forecasting. Mathematics. https://dx.doi.org/10.3390/math11010224

Li, Z., Lin, T., Shang, X., & Wu, C. (2023). Revisiting Weighted Aggregation in Federated Learning with Neural Networks. arXiv. https://dx.doi.org/10.48550/arXiv.2302.10911

Liu, S., Xin, L., Lyu, X., & Ren, C. (2023). Masking-enabled Data Protection Approach for Accurate Split Learning. IEEE. https://dx.doi.org/10.1109/WCNC55385.2023.10118971

Liu, H., Zhang, X., Shen, X., & Sun, H. (2022). Privacy-Preserving Power Consumption Prediction Based on Federated Learning with Cross-Entity Data. IEEE. https://dx.doi.org/10.1109/CCDC55256.2022.10033866

Liu, H., Zhang, X., Shen, X., & Sun, H. (2022). A Fair and Efficient Hybrid Federated Learning Framework based on XGBoost for Distributed Power Prediction. arXiv. https://doi.org/10.48550/arXiv.2201.02783

Liu, H., Zhang, X., Shen, X., & Sun, H. (2022). Privacy-Preserving Power Consumption Prediction Based on Federated Learning with Cross-Entity Data. In 2022 34th Chinese Control and Decision Conference (CCDC) (pp. 181-186). Hefei, China. https://doi.org/10.1109/CCDC55256.2022.10033866

Llasag Rosero, R., Silva, C., & Ribeiro, B. (2023). Forecasting functional time series using federated learning. In L. Iliadis, I. Maglogiannis, S. Alonso, C. Jayne, & E. Pimenidis (Eds.), Engineering Applications of Neural Networks. EANN 2023. Communications in Computer and Information Science, Vol. 1826 (pp. 491–504). Springer, Cham. https://doi.org/10.1007/978-3-031-34204-2_40

Luan, C., Pang, X., Wang, Y., Liu, L., & You, S. (2020). Comprehensive Forecasting Method of Monthly Electricity Consumption Based on Time Series Decomposition and Regression Analysis. IEEE. https://dx.doi.org/10.1109/IAI50351.2020.9262169

Lyu, X., Liu, S., Liu, J., & Ren, C. (2023). Scalable Aggregated Split Learning for Data-Driven Edge Intelligence on Internet-of-Things. IEEE Internet of Things Magazine. https://dx.doi.org/10.1109/IOTM.001.2300053

Ma, S., Xiao, B., Hong, R., Addissie, B., Drikas, Z., Antonsen, T., Ott, E., & Anlage, S. (2019). Classification and prediction of wave chaotic systems with machine learning techniques. Retrieved from https://www.researchgate.net/publication/335159004_Classification_and_prediction_of_wave_chaotic_systems_with_machine_learning_techniques

Makhija, D., Han, X., Ho, N., & Ghosh, J. (2022). Architecture Agnostic Federated Learning for Neural Networks. arXiv. https://arxiv.org/abs/2202.07757

Maksood, F. Z., & Achuthan, G. (2017). Sustainability in Oman: Energy Consumption Forecasting using R. Indian Journal of Science and Technology. https://dx.doi.org/10.17485/IJST/2017/V10I10/97008

Masini, R. P., Medeiros, M. C., & Mendes, E. F. (2020). Machine Learning Advances for Time Series Forecasting. Journal of Economic Surveys, 34(5), 1111-1135. https://onlinelibrary.wiley.com/doi/10.1111/joes.12429

Mundkur, A. (2018). Classification of Respiratory Data: Classifying and analyzing respiratory data to present as feedback towards cultivating habitual diaphragmatic breathing. https://doi.org/10.13140/RG.2.2.26947.22563

Mugunthan, V., Goyal, P., & Kagal, L. (2021). Multi-VFL: A Vertical Federated Learning System for Multiple Data and Label Owners. arXiv. https://arxiv.org/abs/2106.05468

P. Aupke, Seema, A. Theocharis, A. Kassler and D. -E. Archer, "PV Power Production and Consumption Estimation with Uncertainty bounds in Smart Energy Grids," 2023 IEEE International Conference on Environment and Electrical Engineering and 2023 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe), Madrid, Spain, 2023, pp. 1-6, doi: https://ieeexplore.ieee.org/abstract/document/10194894

Pramoditha, R. (2022). Two or More Hidden Layers (Deep) Neural Network Architecture. https://medium.com/data-science-365/two-or-more-hidden-layers-deep-neural-network-architecture-9824523ab903

Qi, T., Zhan, Y., Li, P., Guo, J., & Xia, Y. (2023). Arena: A Learning-based Synchronization Scheme for Hierarchical Federated Learning-Technical Report. arXiv. https://dx.doi.org/10.48550/arXiv.2308.10298

Rehmer, A., & Kroll, A. (2022). The effect of the forget gate on bifurcation boundaries and dynamics in Recurrent Neural Networks and its implications for gradient-based optimization. IEEE. https://dx.doi.org/10.1109/IJCNN55064.2022.9892458

Sharma, N., Puri, V., Mahajan, S., Abualigah, L., Zitar, R. A., & Gandomi, A. (2023). Solar power forecasting beneath diverse weather conditions using GD and LM-artificial neural networks. Scientific Reports. https://dx.doi.org/10.1038/s41598-023-35457-1

Shiwakoti, R. K., Charoenlarpnopparut, C., & Chapagain, K. (2023). Time Series Analysis of Electricity Demand Forecasting Using Seasonal ARIMA and an Exponential Smoothing Model. IEEE. https://dx.doi.org/10.1109/PREE57903.2023.10370319

Sun, H., Shen, L., Chen, S.-Y., Sun, J., Li, J., Sun, G., & Tao, D. (2023). FedLALR: Client-Specific Adaptive Learning Rates Achieve Linear Speedup for Non-IID Data. arXiv. https://dx.doi.org/10.48550/arXiv.2309.09719

Thapa, C., Chamikara, M.A.P., & Camtepe, S. (2020). Advancements of federated learning towards privacy preservation: From federated learning to split learning. In Federated Learning Systems: Towards Next-Generation AI (Studies in Computational Intelligence Series). Springer. https://doi.org/10.1007/978-3-030-53036-5_31

Tin, D., Shahpasand, M., Gharakheili, H., & Batista, G. E. A. P. A. (2022). Classifying Time-Series of IoT Flow Activity using Deep Learning and Intransitive Features. SKIMA. https://dx.doi.org/10.1109/SKIMA57145.2022.10029420

United Nations Framework Convention on Climate Change (UNFCCC). (2015). Paris Agreement on Climate Change (Decision 1/CP.21). Retrieved from https://unfccc.int/process-and-meetings/the-paris-agreement

Wang, H., Zhang, Y., Cheng, Y., Li, Q., Zhao, J., & Li, W. (2023). A Data Privacy Protection Scheme Integrating Federal Learning and Secret Sharing. IEEE. https://dx.doi.org/10.1109/ICPICS58376.2023.10235406

Williams, M. L. (2018). An introduction to Federated Learning. Retrieved from https://mike.place/talks/fl/

Xie, K., Zhang, Z., Li, B., Kang, J., Niyato, D., Xie, S., & Wu, Y. (2022). Efficient Federated Learning With Spike Neural Networks for Traffic Sign Recognition. IEEE Transactions on Vehicular Technology. https://dx.doi.org/10.1109/TVT.2022.3178808

Yu, Z., Sun, Y., Zhang, J., Zhang, Y., & Liu, Z. (2023). Gated recurrent unit neural network (GRU) based on quantile regression (QR) predicts reservoir parameters through well-logging data. Frontiers in Earth Science. https://dx.doi.org/10.3389/feart.2023.1087385

Yule, G. U. (1927). On a method of investigating periodicities in disturbed series, with special reference to Wolfer's sunspot numbers. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character, 226*, 267-298. https://doi.org/10.1098/rsta.1927.0007

Zhang, M., Yu, Z., & Xu, Z. (2020). Short-Term Load Forecasting Using Recurrent Neural Networks With Input Attention Mechanism and Hidden Connection Mechanism. IEEE Access. https://dx.doi.org/10.1109/ACCESS.2020.3029224