
GAZİ ÜNİVERSİTESİ
Gazi
1926

DATA ANALYSIS WITH DISTRIBUTED MACHINE LEARNING METHODS IN SMART CITY APPLICATIONS

Nawaf ABDULLA

DOKTORA TEZİ
BİLİŞİM SİSTEMLERİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

OCAK 2024

Nawaf ABDULLA tarafından hazırlanan "AKILLI ŞEHİR UYGULAMALARINDA DAĞITIK MAKİNE ÖĞRENMESİ YÖNTEMLERİYLE VERİ ANALİZİ" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilişim Sistemleri Ana Bilim Dalında DOKTORA TEZİ olarak kabul edilmiştir.

Danışman: Doç. Dr. Mehmet DEMİRCİ
Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...……

İkinci Danışman: Prof. Dr. Suat ÖZDEMİR
Yapay Zekâ Mühendisliği Ana Bilim Dalı, Hacettepe Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...……

Başkan: Prof. Dr. Osman Ayhan ERDEM
Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...……

Üye: Prof. Dr. Ebru Vesile ÖCALIR AKÜNAL
Şehir ve Bölge Planlama Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...……

Üye: Doç. Dr. Adnan ÖZSOY
Bilgisayar Bilimleri Ana Bilim Dalı, Hacettepe Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...……

Üye: Doç. Dr. İbrahim Alper DOĞRU
Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...……

Üye: Doç. Dr. M. Anıl ŞENYEL KÜRKÇÜOĞLU
Şehir ve Bölge Planlama Ana Bilim Dalı, Orta Doğu Teknik Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...……

Tez Savunma Tarihi: 04/01/2024

Jüri tarafından kabul edilen bu tezin Doktora Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

……..…….

Prof. Dr. Aslıhan TÜFEKCİ

Bilişim Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Bilişim Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

Bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

İmza
Nawaf ABDULLA
04/01/2024

AKILLI ŞEHİR UYGULAMALARINDA DAĞITIK MAKİNE ÖĞRENMESİ YÖNTEMLERİYLE VERİ ANALİZİ

(Doktora Tezi)

Nawaf ABDULLA

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

Aralık 2023

ÖZET

Bu doktora tezi, dinamik tahmin gereksinimleri, kavram sapması ve sürekli değişen akış verileri gibi zorlukların üstesinden gelmeye odaklanarak uyarlanabilir öğrenmenin çeşitli alanlardaki uygulamasını araştırmıştır. Çalışma, Nesnelerin İnterneti (IoT) iş uygulamalarında otonom uyarlanabilir öğrenme yaklaşımlarının önemini vurgulamıştır. İlk bölüm, gecikmeyi en aza indirmek ve performansı artırmak için önerilen işbirlikçi bir sis bulutu mimarisiyle zaman serisi akış verilerindeki kavram sapmasına odaklanmıştır. Deneysel bulgular, temel modellerle karşılaştırıldığında eğitim süresinde ve tahmin hatasında önemli bir azalma olduğunu göstermektedir. İkinci bölüm, çift yönlü LSTM modellerinin pratikliğini ve uyarlanabilir öğrenmenin önemini vurgulayarak derin öğrenme tekniklerinin hava tahmininde uygulanmasını araştırmıştır. Sonuçlar, uyarlanabilir öğrenmenin tahmin hatalarını yaklaşık 45% oranında azaltabileceğini ve tek değişkenli modellerin hesaplama verimliliği faydaları sağladığını göstermektedir. Üçüncü bölüm, kıyı topluluklarındaki taşkın tahminlerini inceleyip iklim değişikliğini dikkate alan kesin tahminlerin kritik doğasını vurgulamıştır. Tahmin doğruluğunu artırmak için çift yönlü LSTM ve uyarlanabilir öğrenmeyi birleştiren uyarlanabilir bir derin öğrenme metodolojisi sunulmuştur. Önerilen sistem temel modellerden 40% daha iyi performans göstermektedir. Son bölüm, kısa vadeli konut enerji tüketimi tahminini ele almış, model doğruluğu, gizlilik ve ölçeklenebilirlik konularını ele almıştır. Uyarlanabilir öğrenmeyi, birleştirilmiş öğrenmeyi ve uç bilişimi entegre eden bir çerçeve önerilenmiş; bu da eğitim süresinin azalmasına, 8% tahmin hatası oranlarına, kullanıcı gizliliğinin korunmasına ve iletişim gecikmesine yol açmıştır. Özetle, bu doktora tezi geniş bir uyarlanabilir öğrenme uygulamaları ve metodolojileri yelpazesini kapsamakta olup, bunların tahmin doğruluğunu iyileştirme, hataları azaltma ve çeşitli alanlarda model performansını artırmadaki etkinliğine odaklanmaktadır.

Bilim Kodu	:	91440
Anahtar Kelimeler	:	Uyarlanabilir öğrenme, makine ve derin öğrenme, birleşik öğrenme, IoT, akıllı şehirler, zaman serisi tahmini
Sayfa Adedi	:	118
Danışman	:	Doç. Dr. Mehmet DEMİRCİ
İkinci Danışman	:	Prof. Dr. Suat ÖZDEMİR

DATA ANALYSIS WITH DISTRIBUTED MACHINE LEARNING METHODS IN SMART CITY APPLICATIONS

(Ph.D. Thesis)

Nawaf ABDULLA

GAZİ UNIVERSITY

INSTITUTE OF INFORMATICS

December 2023

ABSTRACT

This doctoral dissertation investigates the domain of adaptive learning across various applications, to resolve the difficulties presented by dynamic forecasting requirements, concept drift, and ever-changing streaming data. This study examines the shortcomings of traditional mining methods and algorithms in addressing these challenges, underscoring the criticality of autonomous adaptive learning approaches in Internet of Things (IoT) business implementations. The initial section of the thesis centers on concept drift in time-series streaming data, with a specific emphasis on its application in meteorology prediction. To accomplish this, it puts forth a cooperative fog-cloud architecture that exploits the cloud layer's computational capabilities and proximity to IoT devices to deliver optimal performance and minimize latency. The efficacy of this architectural design is substantiated by experimental findings, which illustrate a substantial decrease of nearly 41% in total model training time and a 20% reduction in prediction error compared to baseline models. Subsequently, the thesis explores the application of deep learning techniques to the domain of weather forecasting to predict time series. It investigates the practicality of the bidirectional LSTM model and underscores the significance of adaptive learning via regular model updates. The results suggest that the implementation of adaptive learning can decrease prediction errors by approximately 45%. Additionally, univariate models proved to provide computational efficiency benefits. Next, the thesis examines flood forecasting in coastal communities during its third section, placing particular emphasis on the critical nature of precise predictions considering climate change. It presents an adaptive deep learning methodology that incorporates bidirectional LSTM and adaptive learning to improve the accuracy of predictions while using weakly supervised learning for automatic data labeling and augmentation. The efficacy of the proposed system in predicting floods surpasses that of baseline models by 40%. The concluding section of the thesis inspects the prediction of short-term residential energy consumption, emphasizing the difficulties that arise in terms of model accuracy, privacy, and scalability as time progresses. To resolve these concerns, it proposes a framework that integrates adaptive learning, federated learning, and edge computing. The results indicate that adaptive federated learning performs better than centralized learning. Specifically, it decreases training time by around 80% and forecast error rates by 8%. Furthermore, it safeguards user privacy and minimizes communication latency. In essence, this doctoral dissertation covers a broad spectrum of adaptive learning applications and methodologies, with a particular focus on its efficacy in augmenting prediction accuracy, diminishing errors, and enhancing model performance across diverse domains. These domains include but are not limited to weather forecasting, flood prediction, concept drift in streaming data, energy consumption forecasting in the age of IoT, and dynamic data challenges.

Science Code	:	91440
Key Words	:	Adaptive learning, machine and deep learning, federated learning, IoT, smart cities, and time series forecasting.
Page Number	:	118
Supervisor	:	Assoc. Prof. Dr. Mehmet Demirci
Co-Supervisor	:	Prof. Dr. Suat Özdemir

ACKNOWLEDGEMENTS

I would like to extend my sincere gratitude to my supervisory team, consisting of Professor Suat ÖZDEMİR and Dr. Mehmet DEMİRCİ, for their unwavering guidance, assistance in overcoming the numerous challenges encountered throughout this doctoral program, and instruction on how to become a more proficient researcher.

Many thanks to the thesis monitoring committee members, including Professor O. Ayhan ERDEM, Professor Süleyman TOSUN, and Dr. Adnan ÖZSOY for their constructive feedback and useful suggestions.

Additionally, I would like to express my gratitude to my remarkable companions who endured me on this emotional rolling coaster, with a particular emphasis on my friend, and role model, Sudad ABDUL RAZZAK for his moral support.

Furthermore, I am grateful to my current employer, Energy Pool, for their invaluable assistance throughout the composition of this thesis. In particular, I would like to thank my mentor Nils SIEBERT, our headmaster Alper UĞURAL, and the phenomenal CEO Doğan SÖNMEZ for their patience and understanding.

Ultimately, I would like to extend my deepest appreciation to my cherished father, who consistently encouraged me to strive for and attain greater accomplishments, as well as to my wife, Sara Farag, who stood by my side despite our physical separation and continued to inspire me to complete this endeavor.

This doctorate is dedicated to my late mother, Majedah; I hope that I have made you proud

TABLE OF CONTENTS

Page

ÖZET iv

ABSTRACT v

ACKNOWLEDGEMENTS vi

TABLE OF CONTENTS vii

LIST OF TABLES x

LIST OF FIGURES xii

LIST OF ABBREVIATIONS/ACRONYMS iv

1. INTRODUCTION 1

2. LITERATURE REVIEW 13

2.1. Fog-Cloud Collaboration 13

2.2. Adaptive Learning for Stream Processing 14

2.3. Weather Forecasting 16

2.4. Flood Forecasting 20

2.5. Energy Consumption Forecasting 21

3. METHODOLOGY 27

3.1. Data sets Description and Preprocessing 27

3.1.1. Temperature and weather forecasting 27

3.1.2. Flood forecasting 29

3.1.3. Energy consumption forecasting 30

3.2. Proposed Adaptive Learning Model Architecture 32

3.3. Proposed Deep Learning Models 33

3.4. Evaluation Metrics 38

4. TEMPERATURE AND WEATHER FORECASTING 43

4.1. Introduction 43

Page

4.2. Methods and Materials 47
4.2.1. The collaborative fog-cloud architecture 47
4.2.2. Data split and preparation 48
4.2.3. Proposed adaptive model pipeline 49
4.3. Results and Discussion 50
4.3.1. Comparison of meteorological features 51
4.3.2. Analysis of adaptive machine learning models 53
4.3.3. Analysis of adaptive deep learning models 54
4.3.4. Adaptive model validation using different data sets 57
4.3.5. Local vs. Server computation 59
4.4. Conclusions 61
5. FLOOD FORECASTING 63
5.1. Introduction 63
5.2. Methods and Materials 64
5.2.1. Experimentation setup 67
5.3. Results and Discussion 68
5.3.1. Labeling functions (LF) 68
5.3.2. Analyzing the labeling performance 69
5.3.3. Evaluating the labeling functions 70
5.3.4. The results of the four previously designed scenarios 70
5.4. Conclusions 72
6. HOUSEHOLD ENERGY CONSUMPTION FORECASTING 75
6.1. Introduction 75
6.2. Methods and Materials 78
6.2.1. Proposed adaptive model 78
6.2.2. Federated architecture 79

Page

6.2.3. Experiment setup 82
6.3. Results and Discussion 84
6.3.1. Exploratory data analysis (EDA) 84
6.3.2. Impact of adaptive learning 87
6.3.3. Sensitivity to *N*-clients 88
6.3.4. Sensitivity to *N*-lags 89
6.3.5. Centralized vs. decentralized (federated) architecture 89
6.3.6. Adaptive federated learning: univariate vs multivariate models 90
6.3.7. Comparison with the Literature 91
6.4. Conclusions 92
7. DISCUSSION 95
8. CONCLUSION AND RECOMMENDATIONS 105
8.1. Conclusion 105
8.2. Recommendations 108
REFERENCES 109
ÖZGEÇMİŞ 117

LIST OF TABLES

Table **Page**

Table 2.1. Analysis of recent comparable research on the application of weather forecasting. 19

Table 2.2. Synopsis of the pertinent scholarly works concerning federated learning architecture. 25

Table 2.3. A comparison of recent studies that have examined the application of energy consumption forecasting. 26

Table 3.1. A snippet from the downloaded and modified IMM data set schema. 28

Table 3.2. A description of weather forecasting data sets. 28

Table 3.3. The statistical details contained within the Kerala data set. 30

Table 3.4. The primary details of the residential energy consumption data set for London. 31

Table 3.5. Selected parameters for the LSTM models. 35

Table 4.1. A comparative analysis of ML prediction model. 53

Table 4.2. A comparison of the performance of the baseline and adaptive random forest models. 54

Table 4.3. An evaluation of the three LSTM models that were proposed, with a focus on the effects of adaptive learning and variable quantity on prediction errors (RMSE). 55

Table 4.4. The adaptive bidirectional LSTM model forecasts temperature for a variety of callbacks. 56

Table 4.5. A comparison between the temperature forecasting capabilities of our proposed model and those of cutting-edge LSTM-based models. 56

Table 4.6. MAPE-based comparison of the baseline and adaptive models for the three data sets. 59

Table 4.7. The entire development of this research. 59

Table 4.8. The specifications of the exploited cloud server. 60

Table 4.9. Using a total of ten epochs to compare the efficacy of adaptive models on a local device and the Colab server. 60

Table 4.10. Analyzing the influence of epoch count on the efficacy of the model. 61

Table 5.1. The performance of the labeling models is subjected to analysis. 70

Table **Page**

Table 6.1. Configurations for model testing, training, and simulation. 83

Table 6.2. The 2013 energy consumption statistics for all participating households ... 84

Table 6.3. Comparing adaptive and non-adaptive learning with regard to households energy consumption. 88

Table 6.4. Federated architecture vs. fog-cloud architecture. 90

Table 6.5. Univariate vs. multivariate input-output architecture for adaptive federated learning..... 91

Table 6.6. A comparative analysis of performance with contemporary models..... 92

LIST OF FIGURES

Figure **Page**

Figure 1.1. Smart city applications. 1

Figure 1.2. Data analysis framework proposed in the thesis. 8

Figure 3.1. The flowchart representing the proposed adaptive learning approach. 33

Figure 3.2. The applied stacked LSTM model structure. 36

Figure 3.3. The applied bidirectional LSTM model structure. 37

Figure 3.4. The applied hybrid CNN-LSTM model structure. 38

Figure 4.1. The proposed architecture of the fog-cloud system. 48

Figure 4.2. The proposed temperature forecasting adaptive model. 50

Figure 4.3. The RMSE quantifies the responsiveness of the most frequent meteorological characteristics to the adaptation process. 51

Figure 4.4. The analysis of the IMM four meteorological features using box plots..... 52

Figure 4.5. The trend diagram for the four meteorological features of the IMM. 52

Figure 4.6. When validating the robustness of the model, the RMSE of the baseline and adaptive LSTM is compared across three distinct data sets. 58

Figure 5.1. Labels generation using WSL. 65

Figure 5.2. The flowchart of weakly supervised learning preprocessing methods. 66

Figure 5.3. The proposed flood forecasting model. 67

Figure 5.4. A segment of the labeling functions code. 69

Figure 5.5. Labeling functions' performance analysis..... 69

Figure 5.6. The accuracy of the proposed model and its enhancement steps. 71

Figure 6.1. The proposed adaptive approach to forecasting household energy consumption. 79

Figure 6.2. Federated architecture for forecasting domestic energy consumption. 80

Figure 6.3. The hourly energy consumption of a representative residence. 85

Figure 6.4. The energy consumption histogram of an arbitrary household. 85

Figure 6.5. The lags in the autocorrelation of daily energy usage. 86

Figure **Page**

Figure 6.6. This figure examines the observations, trends, seasonality, and residuals of energy consumption in London households. 87

Figure 6.7. A comparison of the MAE for various customer numbers (participants). ... 89

LIST OF ABBREVIATIONS/ACRONYMS

The symbols and abbreviations used in this study are presented below, along with their explanations.

Abbreviations	Explanation
AI	Artificial intelligence
Bi-LSTM	Bidirectional LSTM
CNN	Convolutional neural network
DL	Deep learning
FF	Flood forecasting
FL	Federated learning
IoT	Internet of things
LF	Labeling function
LSTM	Long-short-term-memory
ML	Machine learning
RNN	Recurrent neural network
WSL	Weakly supervised learning

.

1. INTRODUCTION

Given the accelerated rate of urbanization in the twenty-first century, the notion of a smart city has surfaced as a paradigm shifter. This technological integration improves the quality of life in urban areas. A smart city is a municipality that utilizes information and communication technologies (ICT) to enhance the quality of life for its residents, optimize operations, and increase the efficacy of its resources. Fundamentally, a smart city is a dynamic ecosystem that addresses the complicated nature of contemporary urban environments through intelligent data management and seamless connectivity (Alam, 2021).

One of the prominent applications of smart cities is intelligent traffic management. By employing IoT sensors and data analytics, this system optimizes traffic flow, diminishes congestion, and improves the efficacy of transportation. It facilitates sustainable urban mobility, reduces commute periods, and mitigates traffic-related issues. An additional illustration is Intelligent Energy Management. By integrating IoT devices and data analytics, the organization monitors and optimizes the energy usage of infrastructure, street illumination, and buildings. It reduces carbon emissions, improves energy efficiency, and advocates for sustainable energy practices. Furthermore, intelligent public safety integrates data analytics, surveillance cameras, and sensors to facilitate crime prediction, real-time monitoring, and optimized emergency response. It ensures timely emergency interventions, strengthens law enforcement capabilities, and improves public safety. Figure 1.1 illustrates the applications of smart cities (Alam, 2021).

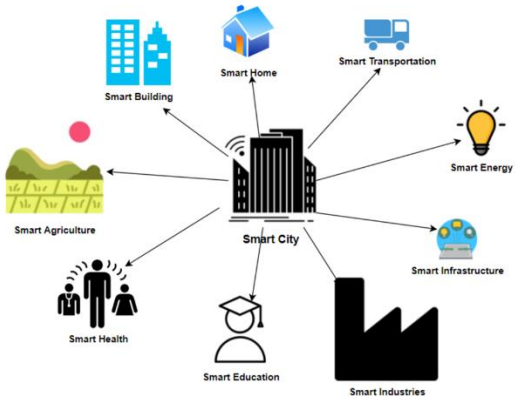


Figure 1.1. Smart city applications.

An essential element of Smart City innovation lies in the methodical incorporation of Internet of Things (IoT) devices throughout the urban environment. Many devices, including intelligent

appliances and sensors, comprise an all-encompassing network that gathers real-time data from diverse aspects of urban existence. The interdependence of Internet of Things (IoT) devices and Smart Cities enables the acquisition of detailed data, contributing to a more nuanced comprehension of urban dynamics. With the transformation of cities into interconnected hubs, the data produced by Internet of Things (IoT) devices assume a critical role, providing decision-makers with valuable information to implement well-informed policies, optimize services, and proactively tackle emergent challenges (Sniderman, Mahto, & Cotteleer, n.d.).

There will be a growing deployment of sensors in many applications, such as smart cities, healthcare, and agriculture, presents significant challenges for IoT systems. These challenges include task management, streaming data processing, and decision-making, among others. Hence, the notions of delivering high performance and minimizing latency in models have a pivotal role in the context of IoT-based smart city applications (Chegini, Naha, Mahanti, & Thulasiraman, 2021). We investigate the core of this thesis, which examines the complex interaction between distributed machine learning and the diverse realm of Smart City applications to unveil the authentic capabilities of urban data. By employing cutting-edge analytical techniques, we aim to positively impact the development of Smart Cities by promoting sustainability, resilience, and an enhanced standard of living for their residents. But before that, we must address the problems related to big data computation since tons of data is generated daily from these ubiquitous IoT devices.

Hence, the introduction of the cloud computing paradigm has emerged as a feasible approach to address the limitations of low-resource IoT devices in performing high computation tasks necessary for IoT-based smart city applications. Cloud computing is commonly portrayed as a worldwide infrastructure that offers substantial computational capabilities and storage capacities. The study by (Alli & Alam, 2020) demonstrates that the abundant resources available in the context of IoT tasks and services are well-suited to meet the demanding performance criteria. According to (Maiti, Shukla, Sahoo, & Turuk, 2018), it has been projected that by the year 2020, there might potentially be an excess of 50 billion interconnected end devices accessing the Internet, resulting in a substantial flow of data. This implies that while the cloud computing paradigm may possess the capability to effectively manage extensive data and process it, the geographical separation between the cloud and IoT devices may result in latency issues. Therefore, it may be argued that cloud computing alone is not a comprehensive solution for IoT-based applications and stream processing.

The need of having a closer proximity technology for IoT devices is essential in addressing time-sensitive IoT applications (Chegini et al., 2021). These applications necessitate a novel paradigm that is characterized by proximity to the user and enhanced computational capabilities. This paradigm, known as fog computing, has been proposed as a solution (Deshmukh & More, 2016). In pursuit of this objective, numerous research studies have embraced fog computing as the most optimal approach. Nevertheless, the issue of determining the optimal paradigm encompasses various aspects. For example, depending exclusively on either cloud computing or fog computing has been found to be ineffective. Consequently, several scholars strive to integrate both computational models under a unified framework known as the Internet of Things (IoT)-fog-cloud ecosystem (Munir, Kansakar, & Khan, 2017).

These two technologies can be distinguished based on many metrics. As an illustration, it can be observed that the cloud infrastructure is situated within the internet, whilst the fog nodes are positioned at the periphery of the network. In contrast to the cloud, the fog computing paradigm operates on a distributed model. The cloud computing concept is characterized by a server that possesses abundant resources and a large-scale storage data center, in contrast to fog nodes which consist of a collection of less resourceful devices. Moreover, the deployment of fog nodes enables the offloading of computational tasks near IoT devices, resulting in a reduction of latency caused by data transfer between them (Munir et al., 2017).

However, it should be noted that these two paradigms are not mutually exclusive, nor can they function independently of each other. Indeed, it can be argued that these entities are intricately interconnected and exhibit a reciprocal relationship, wherein each party derives advantages from the other. In the context of distributed computing, certain functions exhibit superior performance when executed on the cloud layer, while others are more well adapted to the inherent characteristics of the fog layer. The allocation of specific tasks between the front-end (referred to as fog) and the back-end (referred to as cloud) can be subject to dynamic variations dependent on the network status and is specific to the domain (Martin et al., 2017).

The rising demand for processing and analyzing data near edge devices, to enhance data privacy and reduce latency, continues to grow. This resulted in the introduction of another decentralized architecture so-called federated learning (FL), which ensures that training data is processed on individual and edge devices, while they collectively contribute to the development of global machine learning (ML) models (H. B. McMahan, Moore, Ramage, & y Arcas, 2016).

However, FL faces numerous issues that need to be addressed, one of which is the insufficiency or lack of accuracy in the labeled data used for supervised machine learning (ML) and deep learning (DL) algorithms. It is commonly assumed that data labels are adequately annotated for all federated networks. Yet, in fact, data labels may exhibit imbalances, noise, or insufficiency (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014).

Furthermore, it is imperative to address the challenge posed by the ongoing generation of vast quantities of data, commonly referred to as big data (Alli & Alam, 2020). The efficient handling of large-scale data streams is a primary focus within the realm of IoT applications and the corresponding IoT-based decentralized architectures. Therefore, it is important to address this issue by the utilization of automated processing activities, such as Machine Learning (ML) algorithms, as stated by (Chegini et al., 2021). The rationale behind the need for fast and accurate decision-making in handling fast and huge data streams is in the need to minimize or eliminate the necessity for human interaction. However, relying on conventional data mining methods does not align with the dynamic nature of data streams generated by IoT devices. The fluctuating nature of these streams may give rise to diverse patterns over time, hence causing a decline in the accuracy of the model. Therefore, one potential resolution to this issue is to advocate for the implementation of adaptive learning techniques (Zliobaite et al., 2012).

The efficacy of IoT-based smart city applications is dependent upon the diversity and accuracy of extensive data sets acquired from IoT devices. The initial stage involves the detection and transmission of data, although the subsequent and more formidable task lies in extracting actionable insights from the constant flow of large-scale data. This process aims to enhance resource utilization, strategic urban planning, and proactive measures in urban areas (T. Kim, Ramos, & Mohammed, 2017). Hence, it is crucial to use stream processing and analytics effectively to fully benefit from the wide-scale deployment of IoT devices in smart cities. To this end, distributed stream processing frameworks are important enablers for data analytics (Nasiri, Nasehi, & Goudarzi, 2019).

Academic research has examined the correlation between adaptive learning and stream processing in the context of smart city applications. Nevertheless, the utilization of adaptive learning for data stream processing is a highly restrictive technique. Prior research has consistently demonstrated that distributed computing consistently yields superior performance outcomes when compared to the use of either fog or cloud computing in isolation. Furthermore,

research on the processing of streaming data highlights the significance of recognizing concept drift and updating prediction models to effectively address the continuous occurrence of drift in the data stream.

Currently, and based on the available information, there have been no endeavors to investigate the application of adaptive learning for streaming data within the collaborative fog-cloud architecture or federated learning architecture. Most of the work carried out on streaming data and adaptive learning ignores the possibility that we could leverage the high computation resources on the cloud and simultaneously use the proximity of fog nodes to the IoT devices for faster forecasting and interaction.

Aim and research questions

Our aim is to integrate edge and cloud computing to facilitate IoT-based data analytics in some smart city applications. This will be achieved by leveraging edge nodes to minimize data transfer and uphold privacy considerations. In addition to edge-cloud computing architecture, we aim to implement the federated learning architecture and then compare both architectures. However, previous research in edge-cloud architecture and federated learning architecture has made an impractical assumption that the data saved on the local device contains complete annotations with accurate labels, and that the server does not possess any labeled data. In fact, the private data at the local device are more often unlabeled, since annotating data requires both time and domain knowledge, and servers are often hosted by organizations that do have labeled data. Motivated by this, in this thesis, we will study more realistic settings for both edge-cloud and federated architectures. Furthermore, implementing adaptive learning concept to the previously mentioned architectures to ensure that the predictive models are capturing any data or concept drifts happening during the forecasting process. So, the main aim of this thesis could be formed as follows:

We aim to apply adaptive learning to distributed machine learning methods to improve the predictive analytics for various IoT-based smart city applications.

Research questions:

1. Could incorporating adaptive learning into a collaborative fog-cloud architecture improve prediction model accuracy while decreasing real-time data processing latency in IoT-based applications?
2. Is it possible that using a variation of deep learning models for time-series forecasting instead of cutting-edge machine learning models will improve model performance?
3. If weakly supervised learning is used for data labeling and augmentation alongside adaptive learning to handle seasonality in time series forecasting, could we improve the model performance?
4. Could merging adaptive learning to federated leaning architecture improve the predictive model accuracy and result in better privacy-preserving and less training time compared to edge-cloud computing architecture?

Since the scope and domain of this study focus on time-series forecasting using supervised learning methods in distributed architectures, we apply our proposed methodology to three different applications in smart cities that verify this scope, including temperature, household energy consumption, and flood forecasting. The first two represent continuous supervised forecasting, whereas the latter demonstrates discrete supervised forecasting, which was needed to implement the idea of weakly supervised learning. In other words, flood forecasting was selected as it denotes the classification problem in the smart city, where data labeling and augmentation play a big role and bring out a massive problem in literature and industry. It is important to note that the selection of these applications was based on our ability to access free and accessible data sets for future comparison. In summary, each of the three applications utilizes time-series data. Temperature, flood levels, and household energy consumption display distinct patterns and trends as time progresses, thus making adaptive time-series forecasting a suitable methodology.

Overview of approach

The utilization of adaptive artificial intelligence (AI) models is becoming more prevalent in the context of learning from continually evolving stream data, especially in scenarios that are susceptible to frequent changes and trends. These models possess the capability to adapt and modify their parameters in response to real-time data, rendering them well-suited for volatile contexts that need swift and efficient reactions. In addition, these systems possess practical value

in terms of their ability to facilitate real-time monitoring and reaction protocols aimed at minimizing the consequences of security breaches and intrusion attempts.

The effectiveness of smart city applications is contingent upon the diversity and accuracy of extensive data sets acquired from Internet of Things (IoT) devices. Distributed stream processing frameworks play a crucial role in facilitating data analytics inside smart cities. The integration of deep learning and adaptive learning has been a subject of investigation; however, most research has not sufficiently emphasized the utilization of deep learning models to augment the efficacy of the offered models and frameworks.

This study examines the impact of adaptive learning on time-series forecasting and evaluates the role of various long-short-term memory (LSTM) architectures on overall performance and latency. We suggest the utilization of adaptive learning methodologies in distributed machine learning (ML) and deep learning (DL) models to improve performance and enhance prediction accuracy. This is particularly beneficial when handling dynamic data streams provided by Internet of Things (IoT) sensors.

We present three distinct models of Recurrent Neural Networks (RNN), including stacked LSTM, Bidirectional LSTM, and CNN LSTM. The experimental setup encompassed various hyperparameter configurations, such as the number of neurons, the number of units in the dense layer, the dropout value, the selection of activation functions, and the utilization of the default optimizer, Adam. The present study examines a range of Long Short-Term Memory (LSTM) models utilized for the purpose of forecasting univariate and multivariate time series data.

Federated learning (FL) is an algorithmic methodology that enables the training of machine learning models in a decentralized manner. In this technique, individual devices utilize their own local data for the purpose of training the models. The process involves the aggregation of models from several sources into a central server, which then distributes the resulting global model to remote devices for further training. Florida (FL) displays effectiveness in managing non-independent and non-identically distributed data, and it provides benefits in terms of data sharing as compared to centralized training. We propose the integration of adaptive deep learning into the traditional federated learning architecture, aiming to enhance data exchange, privacy preservation, and accuracy. The global model undergoes updates exclusively in the presence of data drift, leading to enhanced model accuracy while minimizing data transfer and

communication. This methodology results in a reduction in the rate of error in forecasting and a decrease in the duration required for training.

This study sets itself apart from other approaches in scholarly literature on energy use by showcasing a robust and prompt response to data drift. Prior methodologies have not adequately considered the recurring occurrence of data drift during the forecasting procedure, resulting in a compromise between the precision of deep learning models and the possible strain placed on the network due to the extensive data needed for repetitive training. Figure 1.2 displays the workflow, architectures, similarities, and differences of the thesis' three conducted studies.

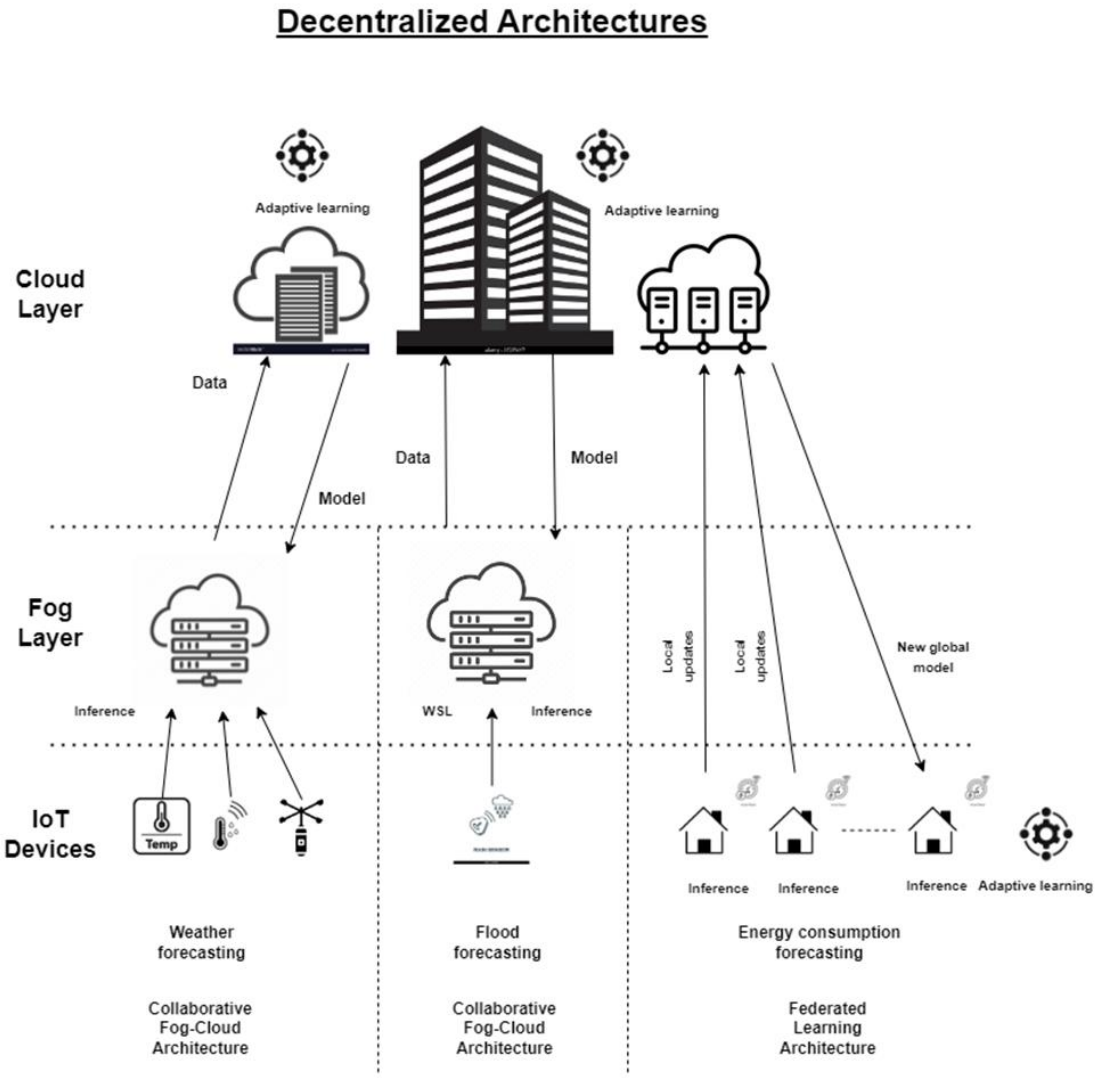


Figure 1.2. Data analysis framework proposed in the thesis.

Research contribution

In this study, we demonstrate our methodology for developing a distributed machine-learning framework tailored for the implementation of IoT-based smart city applications. Furthermore, the recommended framework incorporates the concept of adaptive learning to address the issue of seasonality that arises in quickly evolving data streams. This approach also guarantees that throughput, latency, and model correctness are unaffected, and in fact, enhanced.

1. Compared to the baseline model (non-adaptive model in a local architecture), the adaptive model in the proposed collaborative fog-cloud architecture for weather forecasting reduced prediction error by 20% and training time by approximately 41%.
2. The application of adaptive learning to the bidirectional LSTM-based model reduced the prediction error by 45% compared to the baseline model. In addition, the results show that using only the univariate model results in learning from fewer features and less time and memory requirements for model building and implementation in distributed deep learning architecture.
3. It is shown that using weakly supervised learning as a preprocessing technique for the proposed system can improve the flood prediction accuracy by 40%.
4. Our findings reveal that adaptive federated learning outperforms centralized learning while preserving privacy, reducing communication load, reducing prediction error rate by 8%, and reducing training time by approximately 80%.

Research limitations

Like any other academic investigation, this study possesses inherent limitations. The primary objective of this work is to investigate time-series forecasting in several smart city applications using adaptive and distributed machine learning models. However, it is important to note that the current data sets used for this study may be insufficient as it only spans a period of not more than two years. The length of this span may not be adequate for deep learning methods, such as LSTM models, which require a large amount of data. Empirical evidence supports the notion that the performance of a model improves when it is exposed to a greater number of data points over longer time periods. Therefore, in subsequent studies, it may be advisable to consider utilizing a larger and consequently more comprehensive data set to enhance the representativeness of the findings. Furthermore, the lack of comparable studies utilizing the

identical data set or exclusively addressing a particular application forecasting issue leads to restricted opportunities for comparisons with cutting-edge research. Nevertheless, the authors made a concerted effort to incorporate the prevailing models commonly utilized in the existing body of research within this study. This was done to put together these models against the proposed model, so demonstrating its effectiveness.

Besides, a constraint faced by the researchers was the restricted availability of data sets from prior investigations, which were not publicly accessible. In conclusion, it is anticipated that future study designs would exhibit a greater emphasis on comparative analysis. The authors want to improve and implement the suggested model on readily available and widely accepted benchmarks to evaluate its performance in comparison to state-of-the-art algorithms and methodologies. Moreover, it might be contended that the degree of enhancement is trivial in some applications when we tried the adaptive model in relation to the additional processing capacity, memory use, and communication bandwidth required for the process of retraining when data drift is identified. There are two rationales accounting for the diminished enhancement given by adaptive learning in this experiment. First, the restricted evaluation period in December 2013, for energy consumption study, ensured that data drift did not occur, as household energy usage tends to remain stable over the winter season. Hence, there is no need to update the model utilizing the algorithm we have put forth. The outcome of this led to a slight enhancement in the model's performance. In addition, it is anticipated that the efficacy of adaptability will be showcased as the duration of the testing phase is prolonged to span a broader range of months and varied seasons. Therefore, in the absence of concept drift or data drift, adaptive learning may have little utility in the context of time-series data. Nevertheless, it is important to note that such drifts are frequently observed in time-series applications. Therefore, we argue that the implementation of adaptive learning methods holds value within this context.

Furthermore, up to this point, the research has been carried out using authentic data, albeit in a simulated fashion, employing technologies such as the Apache Spark platform and the TensorFlow federated learning API. The results exhibit promises and deserve validation by deploying these ideas in practical real-world applications. This implementation will add more value to the proposed model and architecture.

Organization

The objective of this thesis is to examine several distributed machine learning techniques and architectures utilized for the purpose of data analysis and forecasting in multiple smart city applications. Therefore, several research articles and studies were conducted throughout the process of writing the thesis. Therefore, this thesis has been created and composed in the format of a complex traditional thesis. The thesis is structured in the following manner. In Chapter 1, the introduction section provides an overview of the context, background, and significance of the topic under investigation. Additionally, it outlines the objectives, research questions, and the strengths and limitations of the thesis. In contrast, Chapter 2 undertakes a comprehensive review of the existing literature and relevant scholarly works. This review serves to highlight the existing gaps in knowledge and to discern the shared themes and discrepancies within the published findings. Subsequently, Chapter 3, which is dedicated to methodology, provides a comprehensive account of the study's applications, data sets, materials, processes, theories, and definitions employed in the research. Subsequently, the succeeding chapters, namely 4, 5, and 6, undertake an individual examination of each smart city application, as presented in respective research studies or articles. Each chapter commences with a concise introduction to the specific issue under consideration, followed by a description of the employed methodology, presentation, and analysis of findings, and ultimately, the formulation of conclusions. Additionally, chapter 7 incorporates a comprehensive discussion of the entire study, while chapter 8, the conclusion, presents the primary findings of the study and offers recommendations for future research.

2. LITERATURE REVIEW

In this section, we aim to analyze the pertinent and substantial research related to the issue under consideration. We will comprehensively present the existing knowledge on the subject matter, as well as highlight the areas that remain unexplored or require further investigation. This will serve as the foundation for justifying the significance and rationale of the thesis, specifically addressing the gap in the existing literature. In addition, we want to discern the shared elements and disparities among the published findings, while also emphasizing the merits and limitations of the prior research studies.

2.1. Fog-Cloud Collaboration

Several studies have examined the benefits of merging both cloud and fog computing powers to realize low latency. For instance, Aburukba et al. (Aburukba, AliKarrar, Landolsi, & El-Fakih, 2020) minimized the latency by employing genetic algorithms to schedule requests of IoT applications. They benefited from the hybrid model of fog-cloud computing to balance the workload and prioritize the processing of urgent requests on the fog layer. Whilst, Munir et al. (Munir et al., 2017) suggested a high performance integrated fog-cloud model that is energy efficient and is able to reduce latency. They demonstrated hypothetically the proposed architecture on different IoT applications such as transportation systems, smart cities, and real-time agricultural data analytics.

Similarly, Sharma and Wang (Sharma & Wang, 2017) proposed a collaborative edge and cloud processing system for wireless networks. Their framework proposition applied to live data analytics. However, the study lacks the practical experiments for the proposed framework. Additionally, Alli and Alam (Alli & Alam, 2020) dealt with the computation offloading between fog and cloud computing using ML and Reinforcement Learning (RL). Using their proposed model, they minimized the latency and energy consumption for smart cities applications.

Other studies have addressed several related topics to the concept of combining cloud computing with other computing paradigms like fog, edge, cloudlet, and so on, such as Hu et al. (Hu, Patel, Sabella, Sprecher, & Young, 2015) who discussed the primary aspects of the market requirements and interests for mobile edge computing. Furthermore, Azam and Huh (Aazam &

Huh, 2015) suggested a layered fog structure to tackle the resources management problem on fog nodes. Based on probability models, they predict the customers' requirements; and accordingly, decisions about resources management are made. Also, Bittencourt et al (Bittencourt, Lopes, Petri, & Rana, 2015) proposed a layered architecture to support and ease mobility of IoT devices. In their framework, each IoT device has a Virtual Machine (VM) instance on the fog node to which is connected. When the device spans the radio spectrum of that fog node, its VM instance is copied and forwarded to the nearest other fog node.

2.2. Adaptive Learning for Stream Processing

Adaptive learning and its relationship with stream processing has been studied in literature. For example, Ku (Ku, 2018) performed a series of experiments to investigate the performance improvement of stream analytics by applying adaptive learning. They compared the speed and accuracy of both traditional ML classifiers and adaptive ones and concluded that better accuracy and speed are achieved by using adaptive learning. Though the study used six data sets and involved intensive experimentation, it was implemented locally using centralized computing.

The majority of the previous research on the problem of adaptive learning on streaming data is for classification problems. For instance, Zliobaite et al. (Zliobaite et al., 2012) developed a model that adapts when it detects a concept drift occurrence. The authors claim that their model, which is based on uncertainty and dynamic allocation, is responsive to any changes that happen within the data set distribution. Some other works have attempted to investigate the classifiers ensembles and different proposed algorithms in the literature to seek out the best and optimal classifiers for online classification (Kuncheva, 2008; Mehta, 2017).

Some researchers have approached the problem from different perspectives. Das et al. (T. Das, Zhong, Stoica, & Shenker, 2014), for instance, argue that the batch size of streaming data being executed affects the throughput and end-to-end latency. Based on that notion, they proposed an effortless but rigid control algorithm which adjusts the batch size automatically when required. On the other hand, Li et al. (H. Li, Wang, Wang, & Zhou, 2017) tackled the issue of imbalanced streaming data via multi-window-based ensemble classification. Their method evaluates the accuracy of the sub-classifiers before new streams arrive and are classified. If the precision value descends under a predefined threshold, the new sub-classifiers are trained. Moreover, Ziobiate and Gabrys (Zliobaite & Gabrys, 2012) addressed the problem of preprocessing in streaming

data by presenting three scenarios involving: adaptive preprocessing, adaptive learning, and adaptive preprocessing and learning together. They argue that incorporating both adaptive preprocessing and adaptive learning contributes to enhancing the performance of ML models.

By contrast, the concept of merging deep learning with adaptive learning was undertaken by several researchers, for example, Tran and Hoang (Tran & Hoang, 2019) carried out a study that proposes an adaptive convolutional neural network model for tracking and re-identifying objects. Their proposed structure ensures higher accuracy with less time and memory complexity. Other good examples are the studies conducted by (Wang, Kubichek, & Zhou, 2018; H. Yu et al., 2020). The former exploits adaptive learning to predict hourly building energy usage, whereas the latter employs it to estimate cardiac motion. Both studies demonstrate better accuracy when using adaptive learning.

In addition to the previous applications of merging together adaptive and deep learning, Pathak et al. (Pathak, Pandey, & Rautaray, 2020) benefits from adaptive learning by designing a deep learning-based framework for topic modeling from big data. Whilst Saxena and Singh (Saxena & Singh, 2022) uses adaptive learning to predict and offload the workload of a dynamic cloud environment. Again, experimental findings show that applying adaptive learning to deep learning algorithms always provides higher accuracy and better resource management.

Overall, these studies highlight the necessity of adaptive learning models to handle streaming data that tend to change their distribution repeatedly. However, the vast majority of these studies have not recommended the exploitation of fog-cloud structures to augment the performance of the proposed models and frameworks. Moreover, the previously discussed studies have not implemented their systems on real-time data stream processing tools like Apache Spark, Flink, Storm, etc. Additionally, the applications used in the previous studies are limited and do not expand the wide range of smart city applications, in specific. Hence, this thesis intends to examine the effect of adaptive learning on data streams prediction for parallel and distributed computing. Also, it aims to study the impact of the distributed architectures such as collaborative fog-cloud architecture and federated learning on these streaming data to improve the overall performance and reduce latency. The targeted smarty city applications in this thesis are weather forecasting, flood forecasting, and energy consumption forecasting. Thus, in the next subsection, we will review the literature based on these three applications.

2.3. Weather Forecasting

Concept drift and adaptive learning have been the subject of a huge and expanding body of research. Various strategies for achieving adaptive learning for machine learning models have been described in previous studies. Adaptive learning for time-series processing applications in smart cities, however, has received little research. However, we divided the literature into subsections based on the many applications of temperature and weather prediction, which include meteorology, sea surface temperature, agricultural, and forest temperature and monitoring. With the growth of sensor-generated weather data, data-driven modeling for meteorological time-series prediction has gotten a lot of attention in recent years. Such data analysis processes have become more effective, flexible, and sound because of recent advancements in machine learning and artificial intelligence paradigms.

The authors (M. Das & Ghosh, 2018) determined that Bayesian networks are the best model for meteorological time-series prediction when compared to other machine learning and statistical models after comparing and examining different computing intelligence methodologies. However, in their studies, they only looked at the multivariate scenario, neglecting the effectiveness of trying out a univariate prediction method. Moreover, they only approached deep learning via artificial neural networks, which represents the basics of such methods. Although the authors in (Y. Li, Zhu, Kong, Han, & Zhao, 2019) implemented a novel attention-based LSTM, they applied it to multivariate problem again. They overlooked the significance of both adaptive learning and univariate prediction.

Similarly, in study (Hewage, Trovati, Pereira, & Behera, 2021), a light-weight data-driven temporal model for a combination of LSTM and CNN was exploited for weather prediction. Then, a contrast between it and other famous regression and statistical models was held. Nonetheless, neither adaptivity nor features variability were tested. Whilst, Hewage et al. (Hewage et al., 2021) proposed a special light-weight LSTM model for time-series weather prediction, exploring both univariate and multivariate prediction. While the obtained results demonstrate better accuracy and efficiency compared to complex traditional models, this model could be further improved if adaptive learning had been exploited.

Sea surface temperature (SST) forecasting is the task of predicting future values of a given sequence using historical SST data (X. Yu et al., 2020). In a recent study in this field, Yu et al.

(X. Yu et al., 2020) proposed and developed an SST prediction model that mixed deep gated recurrent unit with convolutional neural network. They succeeded in providing a more flexible and accurate model for SST prediction, but they disregard the efficiency of using adaptive learning to their proposed model. Moreover, they pass over mentioning whether the applying less features could affect the prediction error rate. Whereas the suggested architecture in another study, Zhang et al. (Q. Zhang, Wang, Dong, Zhong, & Sun, 2017), which combines an LSTM layer and a full-connected dense layer, achieved promising results even when anomaly instances were presented. They even partially adopted the online update concept, leading to confirmed and effective results.

Other studies that follow the same methodology were introduced to the literature and verified the previous conclusions and statements. For instance, in one study (Yuting Yang et al., 2017), the proposed model essentially combines the temporal and spatial information to predict future SST values. Its structure includes one fully connected LSTM layer and one convolution layer. Another example is the work in (J. Liu, Zhang, Han, & Gou, 2018) which analyzes the temporal dependence of marine temperature variation at multiple depths and proposes a new ocean-temperature time-series prediction method. Though the prediction accuracy was improved compared to past experiments, the study lacks the intensive inspection of time-series prediction using deep learning methods.

For making better and sounder agricultural decisions, forecasting seasonal changes of the climate could come in handy. Yet, different agricultural applications, for instance, sowing at the right time and predicting yields, request seasonal projection based on both temporal and spatial features. A good example is (Shin, Kim, & Ha, 2020), where they proposed and implemented a hybrid model combining a global climate model and a regularized extreme learning machine so that the seasonal, daily, mean air temperature at the field scale is forecast. They suggested and designed twenty frameworks, and the results demonstrate that this model slightly improved the prediction accuracy while taking univariate prediction into account. Still, this study could benefit from applying adaptive learning as well as exploring the effect of multivariate prediction. Another work carried out a comparative study for five deep learning models, integrating each one of them with three different pre-processing techniques (Mishra, Bordin, Taharaguchi, & Palu, 2020). Even though it approached the LSTM model and used multivariate predictions, it didn't consider updating the models while training and only two attributes were considered, which are wind power and temperature.

In addition to the previous forecasting applications of statistical and machine learning models, multiple recent works in the literature conducted thorough studies on air pollution and land surface temperature. Specifically, they addressed the influence of the lockdown during Covid-19 on such gases' emissions. Xiao et al. (Xiao, Yang, Fan, Fan, & Al-Qaness, 2020) provided a good example: They developed an LSTM model taking spatiotemporal features into consideration to forecast the PM2.5 concentration daily. They relied on a multi-layer perception model to incorporate the spatial and temporal information with the meteorological time-series data, leading to a weighted LSTM extended model with lower RMSE and MAE. However, this work could be more robust with the approach of adaptive learning to handle the problem of abrupt rises in the PM2.5, especially in the winter due to seasonality.

Likewise and as an extension to the above study, Al-Qaness et al. (Al-Qaness, Fan, Ewees, Yousri, & Abd Elaziz, 2021) designed and developed an adaptive neuro-fuzzy inference system for forecasting air quality considering PM2.5, NO2, CO2, and SO2. While they suggested a hybrid method for optimizing the model above and received better performance compared to other existing algorithms, their model does not perform hourly forecasting. Similarly, other works have been published regarding air pollution and temperature analysis, especially in the times of Covid-19 and lockdown. Almagbile and Hazaymeh (Almagbile & Hazaymeh, 2022) conducted a statistical study analyzing the impact of the lockdown in Amman city on air pollution. While, Sergei et al. (Sergei et al., 2022) attempted to find the correlation between the air and land surface temperatures using satellite mapping. Nonetheless, these studies are more based on statistical approaches than AI and ML.

All in all, this research emphasizes the need of adaptive learning models for dealing with time-series whose distribution changes constantly. Most of the research, however, has not advocated for the use of deep learning models to improve the performance of the proposed models and frameworks. As a result, the goal of this thesis is to investigate the effects of adaptive learning on temperature forecasting as well as the impact of alternative LSTM architectures using different techniques such as MISO and SISO on overall performance and latency.

Table 2.1. Analysis of recent comparable research on the application of weather forecasting.

Study	Addressed problem	Proposed method	Implemented M/DL models	Applied techniques	Data set(s)	Assessment metrics	Pros	Cons
Hewage et al. (Hewage et al., 2021)	Surface weather forecasting	Using two DL-based models and comparing their performance with the classical ML and statistical models	LSTM + Temporal Convolutional Network (TCN)	Multi-input-multi-output (MIMO) + Multi-input-single-output (MISO)	GRIB	MSE	Light, efficient, and up to 12 hrs	Short periods of training and testing (Jan to July, 2018)
Hewage et al. (Hewage et al., 2020)	Surface weather forecasting	Using two DL-based models and comparing their performance with the classical ML models	LSTM + TCN	MIMO and MISO	Local weather station	MSE	Light, efficient, and localized	Short periods of training and testing (Jan to Sep, 2018)
Cifuentes et al. (Cifuentes, Marulanda, Bello, & Reneses, 2020)	Air temperature forecasting	Reviewing the ML models and comparing them	ANN + SVM	MISO	Different data sets	MSE MSE RMSE MAPE	Comprehensive study	Lack of DL models
Al-Qaness et al. (Al-Qaness et al., 2021)	Air quality forecasting	Using an adaptive neuro-fuzzy inference system (ANFIS) with a hybrid optimization method	ANFIS, Slime mould algorithm (SMA), Particle swarm optimizer (PSO)	SISO	AQI data of Wuhan city	MSE MAE MAPE R2	long periods of training (3 years) and testing (1 year), and they analyzed the impacts of the lockdown of Wuhan City on the concentrations of PM2.5 and others.	The forecast horizon is too long (no hourly predictions)
Xiao et al. (Xiao et al., 2020)	PM2.5 forecasting	Using spatial and temporal-aware DL model	LSTM	MISO	PM2.5 concentration, meteorological data on Beijing–Tianjin–Hebei	MAE RMSE P ACC TCC	Making the spatiotemporal correlation between the central site and surrounding sites	The need for adaptivity to address the sudden increase of PM2.5
This study	Temperature forecasting	Using adaptive DL models and comparing their performance to the classical ML models	Stacked LSTM, Bidirectional LSTM, CNN LSTM	MISO and SISO	IMM, Beijing PM2.5, IKA	MAE MAPE RMSE R2	Merging adaptive learning with bidirectional LSTM, Approaching univariate and multivariate problems, Comparing the standalone and cloud computation training times for future collaboration	Not applying MIMO,

2.4. Flood Forecasting

The topic of flood forecasting (FF) has garnered considerable interest in contemporary discourse because of the influence of climate change. Globally, the dynamic climate conditions have led to increasingly erratic flood patterns. In order to efficiently mitigate floods, it is imperative to possess accurate data regarding their frequency, magnitude, and ecological ramifications (Hingmire & Bhaladhare, 2022). Dai et al. (Dai, Tang, Zhang, & Cai, 2021), for example, used Bayesian model combination (BMC-EL) ensemble learning to predict flood depth. Flood intensity classification and K-fold cross-validation increased training set diversity and uniformity. The models' reliability was assessed by the ratio of flood intensity levels. BMC improves ensemble learning model reliability by 16.85\% over BLL. As an extension of this study, Zhang et al. (Z. Zhang et al., 2021) utilized data interpolation and curve fitting to simulate the actual inundation depth. In addition, Linear Regression, Random Forest, Neural networks, Long Short-Term Memory, and Adaptive Boosting are investigated for analyzing, comparing, and evaluating the optimal combinations of flood prediction models, datasets, and scenarios resulting from typhoon presence in Macao.

Mohammed Moishin et al. (Moishin, Deo, Prasad, Raj, & Abdulla, 2021) designed and tested a flood forecasting model using a hybrid deep learning (ConvLSTM) algorithm that combines the predictive power of a Convolutional Neural Network (CNN) and an LSTM Network. The work uses a mathematical Flood Index (IF) derived from the precipitation data set to track water resource depletion. In a flood monitoring system, this Flood Index determines flood duration, severity, and intensity. The new predictive model forecasts daily IF using statistically significant lagged IF and antecedent and real-time precipitation data. Nine rainfall data sets from flood-prone Fiji, which is annually devastated by floods, validate the proposed ConvLSTM model.

Another study by Piadeh et al. (Piadeh, Behzadian, & Alani, 2022) examined the state of real-time flood forecasting modeling in urban drainage systems and its future. Real-time flood forecasting models should include wind flow patterns, soil moisture, fluvial flow, evaporation, and infiltration data along with multiple rainfall assessment sources. Most new RTFF models in UDS used artificial intelligence, so future advances in this technique are expected.

Overall, much research and works in the literature have addressed the problem of flood forecasting, and the main accomplishments are: (i) implementing different machine and ensemble learning models; (ii) predicting many time-steps ahead such as Day+1, +3, +7, and +14; (iii) combining meteorological and hydrological data; (iv) applying data interpolation to sort out “missing data” problem; (v) using suitable imputation techniques, such as linear regression, to fill in data gaps; (vi) using simple deep learning models such as LSTM \& ANN; (vii) applying complex and hybrid models like ConvLSTM, CNNLSTM, and stacked LSTM.

In contrast, this thesis seeks to address unresolved issues in flood forecasting and proposes the following solutions: As a preliminary step, weakly supervised learning is employed for labeling and augmenting data to provide effective imputation techniques for filling in missing data. It is crucial to have accurate data for real-time flood prediction models. Next, a specialized form of RNN known as bidirectional LSTM is implemented to appropriately address the problem of time-series prediction and classification. Then, adaptive learning is applied to the designed model to address the seasonality issue that occurs with such rapid data. For future research, prioritizing processing time and prediction quality is highly advised. To achieve this, distributed architectures for learning - such as collaborative cloud-fog architecture or federated learning (FL) - are highly recommended. The purpose of such a proposal is to provide sufficient lead time for operators to make preventative decisions.

2.5. Energy Consumption Forecasting

Short-term energy consumption forecasting has been a popular topic for many years, with numerous techniques proposed to improve prediction accuracy. Moreover, the selection of the most appropriate methodologies and models is another crucial factor. The surveys (Amasyali & El-Gohary, 2018; Kaytez, Taplamacioglu, Cam, & Hardalac, 2015) analyzed the most pertinent papers on energy consumption forecasting, with a focus on data-driven models. The majority of the reported works focused on non-residential scenarios, with artificial neural networks (ANNs) (Chitsaz, Shaker, Zareipour, Wood, & Amjady, 2015), deep learning (DL) (Aurangzeb et al., 2022), auto-regressive integrated moving average (ARIMA), and support vector machine (SVM) (Nie, Liu, Liu, & Wang, 2012), or other regression methods (Hong, Zhou, Li, Xu, & Zheng, 2020) being the most frequently used models.

Another study provides a comprehensive overview of the methodologies, technologies, and applications of federated learning (FL) in the context of smart cities. Pandya et al. (Pandya et al., 2023) conduct a comprehensive analysis of FL in the context of smart cities, encompassing societal, industrial, and technological factors that contribute to its advancement. Their paper examines several smart city applications that incorporate FL, including smart transportation systems, smart healthcare, smart grid, smart governance, smart disaster management, smart industries, and UAV for smart city monitoring. Additionally, the authors discuss the concerns related to data leakage and privacy in the context of FL for smart city applications. It also focuses on the current and upcoming projects related to FL in the context of smart cities. Notable projects include those funded by the European Union (EU), the Defense Advanced Research Projects Agency (DARPA), as well as various industry and research initiatives. The authors conclude by examining the research challenges and prospects for the technological advancement of FL in the context of smart cities.

Also widely employed in load forecasting is deep learning. Mocano et al. (Mocanu, Nguyen, Gibescu, & Kling, 2016) proposed a method based on the conditional restricted Boltzmann machine is one of the most pertinent works in this context (CRBM). The authors provide evidence to support the superiority of their method over the currently employed techniques of Artificial Neural Networks (ANN) and Support Vector Machines (SVM). Similarly, Shi et al. (Shi, Xu, & Li, 2017) proposed a polling-based deep recurrent neural network (RNN) to prevent (or severely restrict) overfitting. The method outperformed conventional techniques including ARIMA, support vector regression, and RNN. The findings from both investigations were promising; nonetheless, the computational demands of deep learning techniques often exceed the capabilities of edge devices with constrained computing resources.

Fekri et al (Fekri, Patel, Grolinger, & Sharma, 2021) is among the first works in this field to implement and evaluate LSTM. The research conducted by the authors showcases the higher performance of Long Short-Term Memory (LSTM) networks in comparison to standard back-propagation Artificial Neural Networks (ANNs). This superiority can be attributed to LSTM's enhanced capacity to identify and capture long-term temporal correlations. Also, Alonso et al. (Alonso, Nogales, & Ruiz, 2020) is a contemporary LSTM-based method. The suggested method has a high level of adaptability, as it is capable of effectively forecasting

energy consumption even in cases when the LSTM model is employed for load prediction in residential houses lacking historical data within the training set. Nevertheless, the attainment of such adaptability can solely be guaranteed by the utilization of a substantial quantity of training data, hence becoming the procedure computationally demanding.

Although the previously mentioned machine-learning-based solutions differ in many ways, they all share one architectural characteristic: To train a global model in a centralized manner, it is necessary to establish a centralized body responsible for gathering energy consumption data from clients. This characteristic is known as centralized architecture, and we will use it as a benchmark for our proposed model and framework. Based on existing literature, two recent scholarly articles have utilized federated learning techniques for load forecasting specifically in edge computing environments (Taïk & Cherkaoui, 2020) and (Savi & Olivadese, 2021).

In relation to load forecasting, a recent study, Yang et al. (Yang Yang, Wang, Zhao, & Wu, 2023), developed a novel algorithm, named VMD-FK-SecureBoost, to address the challenge of sharing data while preserving the privacy of participants. The algorithm integrates three techniques: variational mode decomposition (VMD), federated k-means clustering (FK), and SecureBoost, to enhance prediction accuracy. The utilization of VMD enables the decomposition of the initial dataset into smaller subsequences, hence facilitating the extraction of implicit characteristics and enhancing the accuracy of predictions. Then, FK is employed to recombine the subsequences into clusters with shared characteristics. Finally, SecureBoost is utilized to protect privacy during federated learning. The experimental results demonstrate that VMD-FK-SecureBoost outperforms other algorithms, such as XGBoost and SecureBoost, with the lowest mean absolute percentage errors (MAPEs) for one-step-ahead forecasting in Texas and Newcastle CBD. Therefore, this algorithm has the potential to enable data sharing while preserving the privacy of participants. However, their study differs from ours in terms of the utility of the data sets and machine learning algorithms used. That is, alternative machine learning methods were employed apart from LSTM, and a distinct data set was utilized, which was not accessible during the composition of our work. Thus, maintaining a comparison with them poses a significant challenge.

In a problem resembling load forecasting, Thorgeirsson et al. (Thorgeirsson, Scheubner, Fünfgeld, & Gauterin, 2021) employs an extended version of the federated averaging

algorithm to train probabilistic neural networks and linear regression models in a manner that is both communication-efficient and privacy-preserving. The experiments illustrate the superior performance of probabilistic prediction algorithms compared to deterministic prediction models when evaluated using appropriate scoring rules. Furthermore, their findings demonstrate the potential of federated learning to improve conventional driver-specific learning methods. The implementation of probabilistic predictions allows for the adjustment of safety margins based on the attainability of the destination. This leads to an improved driving range and reduced travel time. The results emphasize the potential of using federated learning and probabilistic prediction models to enhance driving efficiency and safety. Since their study relies on probabilistic forecasting for electric vehicles (different applications), it cannot be directly assessed with the proposed model for our study.

In addition to load forecasting, Liu et al. (F. Liu et al., 2022) creates the first benchmark for detecting time series anomalies in federated learning frameworks. Five-time series anomaly detection algorithms, four federated learning frameworks, and three-time series anomaly detection data sets are all part of the benchmark. Through the experiments, they looked at how federated learning works, what happens when data is not spread out evenly, and how well different federated learning frameworks and time series work together. These analyses help researchers choose the right time series anomaly detection algorithms and federated learning frameworks when doing time series anomaly detection within federated learning frameworks. A concise overview of the previously described research is provided in Table 2.2 to offer readers a quick glimpse. The table shows that while there are several studies applying deep learning (DL) to the problem of energy consumption forecasting in IoT, they have not enhanced their approaches by using federated learning (FL) and adaptive learning (AL) together.

Table 2.2. Synopsis of the pertinent scholarly works concerning federated learning architecture.

Ref.	Year	Preview	Scope				
			IoT	Energy consumption	DL	FL	AL
(Pandya et al., 2023)	2023	Explore FL-integrated smart city applications.	✓		✓	✓	
(Mocanu et al., 2016)	2016	Show that the conditional-constrained Boltzmann machine surpasses ANN and SVM load forecasting.	✓	✓	✓		
(Shi et al., 2017)	2017	Suggest a polling based deep RNN reduce overfitting in load forecasting.	✓	✓	✓		
(Fekri et al., 2021)	2021	Show that LSTM finds long-term temporal correlations better than standard back- propagation ANNs.	✓	✓	✓		
(Alonso et al., 2020)	2020	Adaptable approach accurately predicts energy usage, even with no past data in the training set for households using LSTM model.	✓	✓	✓		✓
(Taïk & Cherkaoui, 2020)	2020	Use univariate (lags) LSTM for energy consumption forecasting.	✓	✓	✓	✓	
(Savi & Olivadese, 2021)	2021	Implement multivariate LSTM models that are locally trained by users using their own historical energy usage samples using Federated Learning and Edge Computing.	✓	✓	✓	✓	
(Yang Yang et al., 2023)	2023	Develop the VMD-FK-Secure Boost algorithm that addresses data sharing while protecting participant privacy.	✓	✓	✓	✓	
(Thorgeirsson et al., 2021)	2021	Train probabilistic neural networks and linear regression models using an improved version of the federated averaging approach to maximize communication efficiency and privacy.	✓	✓	✓	✓	
(F. Liu et al., 2022)	2022	Create the first federated learning framework time series anomaly benchmark.	✓		✓	✓	
This study	2023	Add adaptive learning to LSTM for energy consumption utilizing univariate and multivariate models for FL and centralized systems.	✓	✓	✓	✓	✓

In this thesis, we utilize LSTM and distribute the training load across multiple edge nodes for energy consumption as applied in similar previous studies (Savi & Olivadese, 2021; Taïk & Cherkaoui, 2020). Despite the fact that both (Taïk & Cherkaoui, 2020) and our study use univariate LSTM, i.e. the only input feature to the model is previous timesteps of energy consumption, we employ adaptive learning in addition to federated learning to enhance model forecasting performance. Furthermore, our work differs from (Savi & Olivadese, 2021) by using both univariate and multivariate input architecture, and then implementing adaptive learning to improve the model's overall performance; that is, the model's accuracy and efficiency. The results demonstrate that our proposed model outperforms both models in terms of Root Mean Squared Error (RMSE).

Table 2.3. A comparison of recent studies that have examined the application of energy consumption forecasting.

Study	Problem	Proposed architecture	LSTM type	Input-Output architecture	Data set(s)	Metrics	Strengths	Limitations
(Taïk & Cherkaoui, 2020)	Household electrical load forecasting	Edge Computing (EC) and FL	Stacked (2-hidden-layered)	Single-input-single output (SISO)	200 houses From Texas, USA.	RMSE, MAPE	The first application of federated learning to household load forecasting	Training inputs narrowed down to just energy consumption
(Savi & Olivadese, 2021)	Household short-term energy consumption forecasting	EC and FL	Stacked (2-hidden-Layered)	Multiple-input-single output (MISO)	Smart meter energy consumption data in London households	RMSE, Training time, Trans-mitted data	Strategy based on customer clustering, extra features related to weather and calendar as input.	No adaptive learning or other LSTM model types
This thesis	Household short-term energy consumption forecasting	EC and FL	Stacked (3-hidden-layered) + Bidirectional LSTM	SISO and MISO	Smart meter energy consumption data in London households	MAE, RMSE, Training time	It implements adaptive learning and other LSTM model types	It lacks deep investigation into other federated learning aspects, including communication overhead

3. METHODOLOGY

The data set gathering, and preparation are described in this section. Also, the system model and its workflow, and the assessment metrics are also explained.

3.1. Data sets Description and Preprocessing

Since this thesis approaches different smart city applications, we have several different data sets that belong to the same application or vary from one application to another. Accordingly, we divide this section into three subsections based on the three smart city applications that we are interested in.

3.1.1. Temperature and weather forecasting

The targeted domain for this thesis was IoT-based time-series forecasting, specifically the use of meteorology. We chose meteorology because we planned to test our theory using streaming data. Furthermore, one of the purposes of the work was to investigate and choose the best adaptive prediction model for this data set. As a result, the Meteorology observation station data collection from the Istanbul Metropolitan Municipality (IMM) Open Data Portal was adopted as the case study. We then applied the models to two more publicly accessible data sets, the Korean Meteorological Administration (KMA) and the Beijing PM2.5 data sets, to confirm their performance and efficacy.

Information is gathered hourly from sensors situated across Istanbul for the first data set, IMM, which includes temperature in Celsius, humidity in percent, wind (km/hr), and precipitation in cm. The lowest, maximum, and average values of the previously specified attributes are included in this data. The fire department in Turkey is the source of this data set. This project began on November 17, 2020, and the data set has been created and updated monthly since January of that year. The data is delayed 45 days and updated regularly. At the time of writing, the most recent update was made in May of 2021.

Because meteorological conditions fluctuate seasonally, it is consistent with the concept of adaptive learning. Furthermore, its contents may be modified and used to univariate or multivariate time-series prediction issues. Table 3.1 shows an overview of the updated data

set structure and instance. We may concatenate data from successive months of a given district from separate files (roughly 16 months), preprocess them, and then save them in a single CSV or txt file to increase the sample number for a certain place.

Table 3.1. A snippet from the downloaded and modified IMM data set schema.

Date Time	Temperature (Celsius)	Humidity (100%)	Wind (km/hr)	Precipitation (cm)
Feb. 5 2021 23:00	9.756	96.005	0.455	0.499
Feb. 6 2021 0:00	9.349	98.449	0.656	1.052
Feb. 6 2021 1:00	9.069	99.160	0.619	4.641
Feb. 6 2021 2:00	9.927	99.275	1.217	6.188
Feb. 6 2021 3:00	10.348	99.300	1.313	9.550

The second data set, on the other hand, is for bias correction of the Korea Meteorological Administration's LDAPS model's forecast of next-day maximum and minimum air temperatures over Seoul, South Korea. Summer data from 2013 to 2017 is included in this data set. The input data is mostly made up of next-day forecast data from the LDAPS model, current-day maximum and minimum temperatures, and geographic auxiliary variables. This data has two outputs (maximum and lowest air temperatures for the next day). For the years 2015 to 2017, hindcast validation was carried out (Cho, Yoo, Im, & Cha, 2020).

The third exploited data set, however, was collected between Jan 1st, 2010, and Dec 31st, 2014. This data set comprises temperature, precipitation, pressure, wind in MPH, and PM2.5 concentration (Liang et al., 2015). Table 3.2 shows some information related to all data sets taken into consideration.

Table 3.2. A description of weather forecasting data sets.

Features	IMM	KMA	Beijing PM 2.5
Data Set Characteristics	Multivariate	Multivariate	Multivariate
Attribute Characteristics	Real	Real	Integer and Real
Associated Tasks	Regression	Regression	Regression
Number of Instances	12,000	7,750	43,823
Number of Attributes	24	25	13
Missing Values?	NO	YES	YES

The topic of Multi-Input-Single-Output (MISO) time-series forecasting is the focus of this research alongside Single-Input-Single-Output (SISO). So far, we've just chosen to deal with

temperature, humidity, wind speed, and precipitation as inputs and the target (output) is the temperature of the next hour. Naturally, the collected data sets are sampled as the following: 90% of the former and historical data points are exploited for training and validating purposes, whereas the last recent 10% is used for testing. Furthermore, the window length of forecasting (forecast horizon) is 24 hours (day+1) while the look-back parameter varies from 6, 12, 18, to 24 timesteps.

Before sampling the data sets, the following are the steps involved during the data preparation and engineering:

1. We deleted the unnecessary columns and simply maintained the average temperature data for each hour, omitting the minimum and maximum temperature values.
2. We chose just the observations from the AHL-BAKIRKOY district, where ICING sensors are placed, for the purpose of simplicity. The sample selection technique was purely random.
3. We selected the temperature characteristic for assessing the performance of the proposed prediction model and suggested architecture because this article focuses on univariate time-series prediction.
4. We didn't have to do any data transformations while dealing with the one-feature scenario (i.e., univariate). However, for multivariate predictions, the Min-Max normalization technique was adopted to unify the values of whole features between 0 and 1.
5. Since the last two data sets contain missing values, a simple solution was to remove the samples containing them provided that the rows containing missing values are less than 5%.

3.1.2. Flood forecasting

Typically, meteorological, and hydrological characteristics are employed for flood forecasting. For example, precipitation index and flood probability. This study aims to classify water inundation or the detection of flood events. Because the most common rainfall measurement is the total rainfall depth over a given period, expressed in millimeters (mm), this study used the Kerala data set as a benchmark for the proposed model. The Kerala data set has monthly rainfall information from 1900 to 2018, including flood occurrences. It can be used to predict future flooding by analyzing rainfall patterns. The data is stored in a CSV

file (Pramanick et al., 2022). Table 3.3 illustrates the yearly statistical information on the Kerala data set.

Table 3.3. The statistical details contained within the Kerala data set.

Statistical measure	Yearly Rainfall
Count	118
Mean	2925.41
STD	452.17
Min	2068.80
25%	2613.53
50%	2934.30
75%	3170.40
Max	4473.00

3.1.3. Energy consumption forecasting

This application makes use of a data set titled ``SmartMeter Energy Consumption Data in London Households," which was obtained from the London Datastore website (DataStore, 2018) and imported into our data file. Between November 2011 and February 2014, 5,567 samples were collected from London households as part of the UK Power Networks-led Low Carbon London project. Readings were taken every 30 minutes. The trial participants were selected to be representative of the Greater London population. Energy consumption in kWh (per half hour), date and time, and a unique household identifier are included in the data set. Unzipped, the CSV file contains about 167 million rows and is approximately 10 GB in size.

Extraction, Transformation, and Load (ETL)

Python and related libraries were used to manipulate the imported raw data. For instance, we used Pandas for data extraction, exploration, cleaning, and loading. Also, the holidays library was used to extract the UK holidays for specific years, and the NumPy library was used for some arithmetic operations conducted on the data set. The process of the ETL is explained as follows: In the beginning, we extracted all the features from the CSV file, which included the seven attributes above. Then, we narrowed our focus to the most pertinent characteristics for this study, including energy consumption, a unique household identifier, a date, and a time. Then, we divided the households that had samples and readings for the entire year of 2013. We made this decision because we intended to compare our findings to those of the previous related study (Savi & Olivadese, 2021). As a result, we down sampled the sampling rate from every half-hour to every hour for the purposes of comparison. The

modified data set was finally saved as a CSV file. The primary characteristics of the chosen data set are displayed in Table 3.4.

Table 3.4. The primary details of the residential energy consumption data set for London.

Characteristic	Description
Number of samples	241,714
Missing values	none
Start date	01-01-2013
End date	31-12-2013
Number of features	3
Features	household id, energy consumption, and timestamp
Data types	int, float, date-time
Data set size	31.4 MB

Supplementary features

This section explains the process by which the data set was augmented with supplementary attributes, such as the timestamp of each sample's acquisition and the corresponding hourly energy consumption measurement. The study utilized three calendar attributes, namely the day of the year, weekday, and hour of the day, to investigate and understand the daily, weekly, and seasonal patterns. The attribute denoting the day of the year was expressed as an integer ranging from 0 to 365. Also, the weekday variable was represented as an integer within the range of 0 to 6. The hour variable was obtained by extracting information from the timestamp and afterward represented as a number ranging from 0 to 23. Subsequently, the variables representing the hour and weekday were manipulated and transformed into sine and cosine functions to effectively capture the cyclical patterns inherent in the progression of hours and weekdays.

In addition, lag variables representing past energy consumption, temperature, and humidity were utilized, all of which are relevant to calendar functionality and meteorological conditions. The lag variable was constructed by utilizing the energy consumption value from the same hour of the previous day (referred to as lag-24). The temperature and humidity forecasts of the prediction hour are utilized as input features for the LSTM model.

All in all, the final data set we composed consists of the input features listed below, and the output target is the next hour's energy consumption. The ultimate data set was used in two ways: (i) univariate input forecasting, where the input data only relies on one feature, which is the previous energy consumption in the past hours; (ii) multivariate input forecasting,

where the LSTM model is fed with more information than the previous energy consumption, including calendar data, weather data, etc. The input features and target list are:

1. Timestamp
2. Hour (sin and cosine)
3. Weekday (sin and cosine)
4. DayOfYear
5. Temperature
6. Humidity
7. Lag-24 (energy consumption 24 hours ago)
8. Energy consumption (next hour - target variable)

3.2. Proposed Adaptive Learning Model Architecture

Adaptive AI models have gained interest in learning from continuously evolving stream data, especially when it is prone to frequent changes and trends, such as concept drift. Conventional mining techniques and algorithms are insufficient to solve this problem, as the model's performance degrades even with stationary data. Adaptive AI models possess the ability to dynamically readjust based on real-time feedback, making them suitable for unpredictable environments requiring fast response. They are also useful for real-time monitoring and response mechanisms to mitigate the impact of security breaches and intrusion attempts.

Smart city applications depend on the variety and correctness of large amounts of data collected from IoT devices. To fully benefit from wide-scale deployment of IoT devices in smart cities, distributed stream processing frameworks are important enablers for data analytics. The relationship between adaptive learning and stream processing for smart city applications has been studied in the literature, with most prior research focusing on classification problems.

Several researchers have explored the idea of combining deep learning with adaptive learning, such as Tran and Hoang (Tran & Hoang, 2019), Wang et al. (Wang et al., 2018), and Yu et al. (X. Yu et al., 2020). These studies highlight the significance of using adaptive learning models when dealing with time-series whose distribution is dynamic. However, most of these studies have not promoted deep learning models to enhance the effectiveness

of the proposed models and frameworks. Having said that, This thesis investigates how adaptive learning influences temperature forecasting, and how different long-short term memory (LSTM) architectures (such as multi-input-single-output (MISO) and single-input-single-output (SISO)) affect overall performance and latency. The general adaptive AI framework that can be used for the forecasting problems discussed in the next chapters and sections is depicted in Figure 3.1.

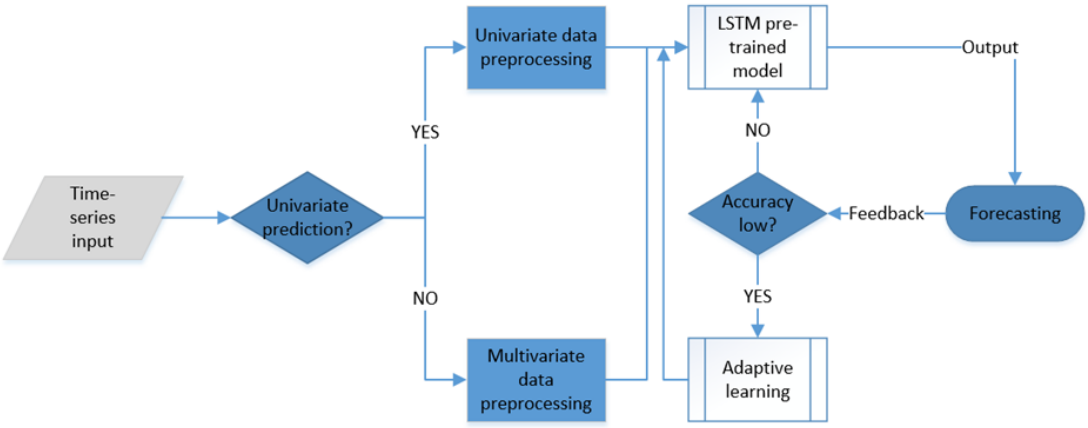


Figure 3.1. The flowchart representing the proposed adaptive learning approach.

3.3. Proposed Deep Learning Models

We propose using adaptive learning approaches in both distributed ML and DL models to achieve finer performance, enhancing prediction accuracy, to cope with the challenge of constantly changing data streams supplied by IoT devices. Certainly, there is a large body of literature on machines and deep learning algorithms that details the improvements over previous techniques. However, there has been little study on evaluating and comparing the models' performance.

Machine learning (ML) is designed to enable a system to acquire information from historical or current data and utilize that knowledge to make informed predictions or judgments in the future. Deep learning (DL) is a sub-field within the broader domain of machine learning (ML) that enables the training of computer models characterized by multiple layers of processing. These models are designed to acquire and internalize data representations. The various layers of features in deep learning are automatically identified and combined to get the desired outputs. In contrast to ML, DL offers a significant advantage in the form of automated feature extraction, which eliminates the need for manual effort in building feature

representations. DL has received considerable acclaim in many fields such as computer vision, natural language processing, and bioinformatics. In recent years, both academia and industry have expanded the utilization of DL techniques to encompass a broader range of applications, including those within the Internet of Things (IoT) domain. IoT networks generate a substantial volume of data, necessitating alternative approaches to data collection, storage, and processing that can effectively handle this magnitude. DL models typically become more successful through the use of such big data (Y. Liu, Zhou, Yang, & Wang, 2023).

In this work, we introduced three types of Recurrent Neural Networks (RNN) models called stacked LSTM, Bidirectional LSTM, and CNN LSTM. Although the three models belong to the same category, they are quite different in their inner structure. Nonetheless, we practically implemented the same hyper-parameters settings on the three structures for the sake of equality, except for the number of hidden layers, which differs from one model to another, and the reason is explained next.

The hyperparameters settings used in the experimentation are shown in Table 3.5. . As illustrated from the table, the number of neurons is set to 50, because it was noticed that decreasing the number leads to lower accuracy, and increasing it further complicates the structure, and hence, requires more computation time for insignificant improvement on the accuracy. In addition, the number of units in the dense layer is set to 1 since the output is always one variable (e.g., temperature). Besides, to avoid the overfitting problem, the dropout value is assigned to 20% as a starting point because it is accepted as the best trade-off between preventing overfitting and keeping good accuracy. Though we raised it to 50%, we did not achieve any additional enhancement, so we maintain the old value (i.e., 20%). Furthermore, we implemented various activation functions such as Sigmoid, hyperbolic tangent, and Relu functions. But the best results that we received were from using the Relu function. As our model performs forecasting, the popular optimizer for such cases is used, which is ADAM. The Adam optimizer produces results that are superior to those of conventional optimization methods, take less time to compute, and need fewer tuning parameters. Hence, Adam is suggested as the default optimizer for our application. Equally, the batch size selected for this work is 32, which is an acceptable default value in the literature. However, when we enlarged the size of the batch from 32 to 64, 128, and 256, we discovered that the performance respectively degraded, especially timewise.

Table 3.5. Selected parameters for the LSTM models.

Hyper-parameters	Applied value/function
Neurons number	50
No. of units in dense layer	1
Dropout	20%
Activation function	Relu
Loss function	Mean Square Error
Optimization function	ADAM
Epochs number	50
Batch size	32

Directly, a number of LSTM models for univariate and multivariate time series forecasting are explained. To begin with, a Stacked LSTM model is created by stacking multiple hidden LSTM layers one on top of the other. An LSTM layer requires a three-dimensional input, and by default, LSTMs produce a two-dimensional output because of the sequence's conclusion. With the return sequences=True option on the layer, the LSTM can produce a value for each time step in the input data, enabling us to use the 3D output of a hidden LSTM layer as an input to the following layer. As a rule of thumb, a single layer works fine for simple systems, whereas a double layer and more is better for complex ones. Moreover, more nodes (neurons) could improve the accuracy while fewer of them could result in an underfitting problem. Figure 3.2 shows the structure of the proposed stacked LSTM model. We began with one layer (vanilla LSTM), then stacked it with more and more layers until we achieved desirable outcomes. As shown in Figure 3.2, the employed stacked LSTM model in this work is a 4-layer one.

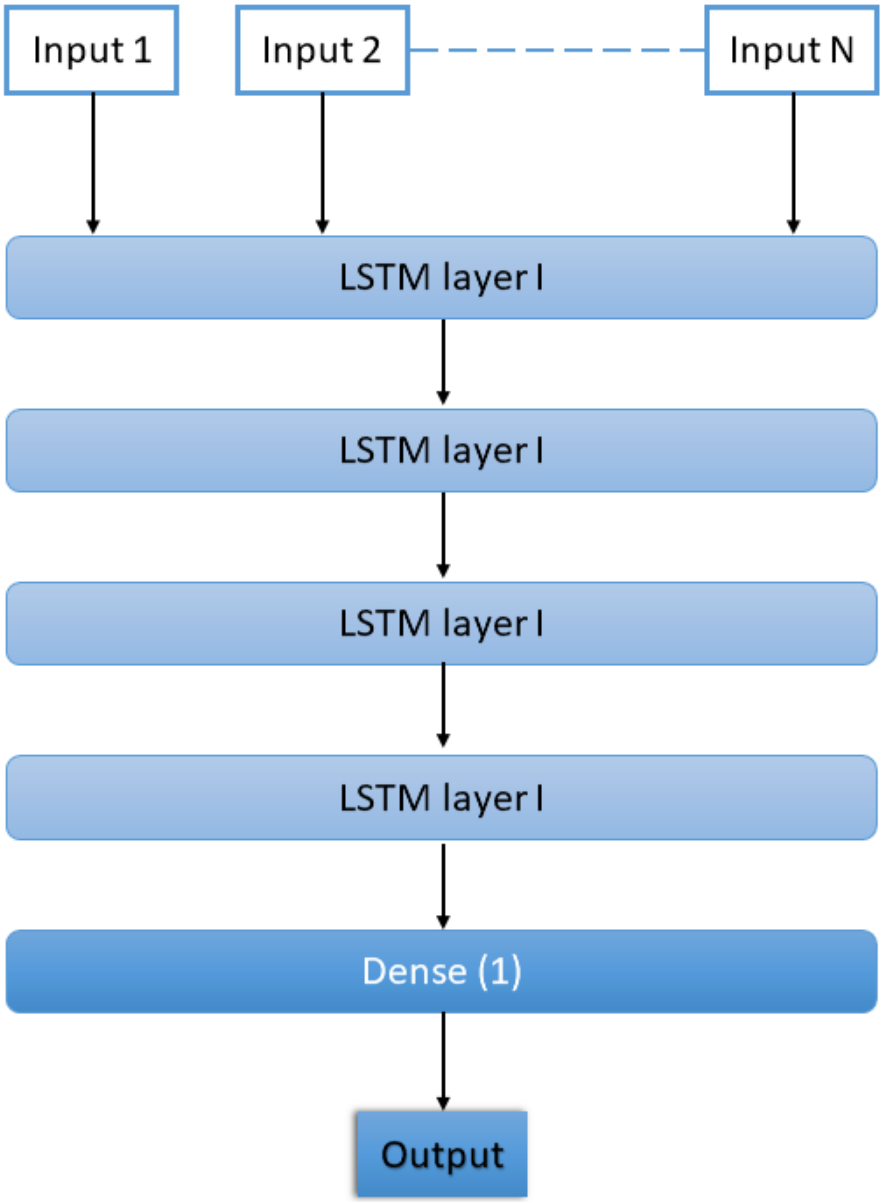


Figure 3.2. The applied stacked LSTM model structure.

On some sequence prediction problems, however, allowing the LSTM model to learn the input sequence forward and backward and concatenate both interpretations can be advantageous. This is called a bidirectional LSTM model. Bidirectional LSTM is a type of LSTM that can be used to increase model performance in sequence prediction tasks. Bidirectional LSTM trains two instead of one LSTM on the input sequence in problems where all time steps of the input sequence are known: the first based on the original input sequence, and the second based on a reversed replica of the original input sequence. This can give the network more context and enable it to learn the problem faster and more thoroughly. As we reach better accuracy with the simplest structure of bidirectional LSTM

compared to stack one, we envisage that complexity is not always the solution. Yet, we increased the layers of it but acquired minor enhancements. So, we decided on the simplest structure that gives us desirable performance because it requires less time for model construction and, subsequently, less time and memory computations. See Figure 3.3 for a detailed demonstration.

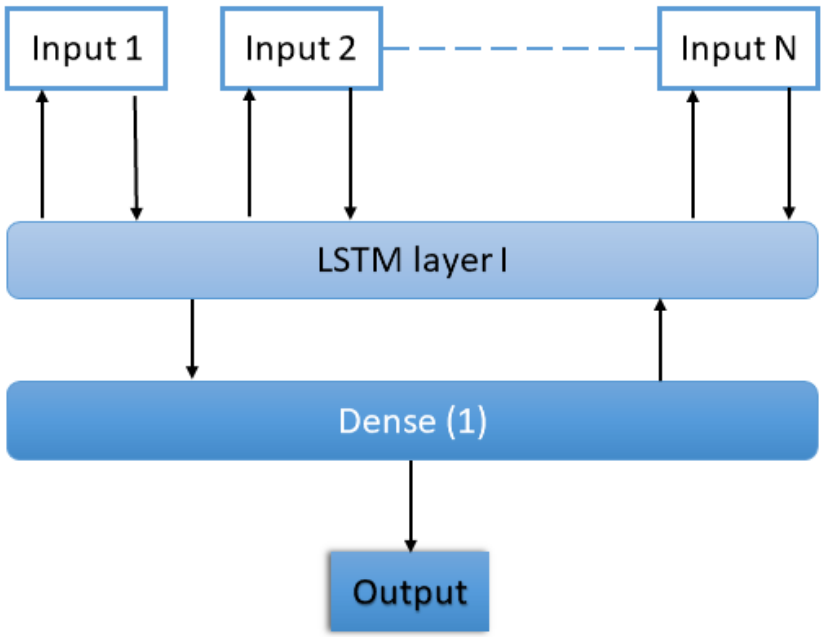


Figure 3.3. The applied bidirectional LSTM model structure.

On the other hand, a CNN, or convolutional neural network, is a sort of neural network designed to operate with two-dimensional picture input. Automatic feature extraction and learning from one-dimensional sequence data, such as univariate time series data, can be quite effective with CNNs. In a hybrid model with an LSTM backend, a CNN model can be used to understand sub-sequences of input that are delivered as a sequence to an LSTM model. A CNN-LSTM is the name for this hybrid model. Figure 3.4 displays an overview of its structure.

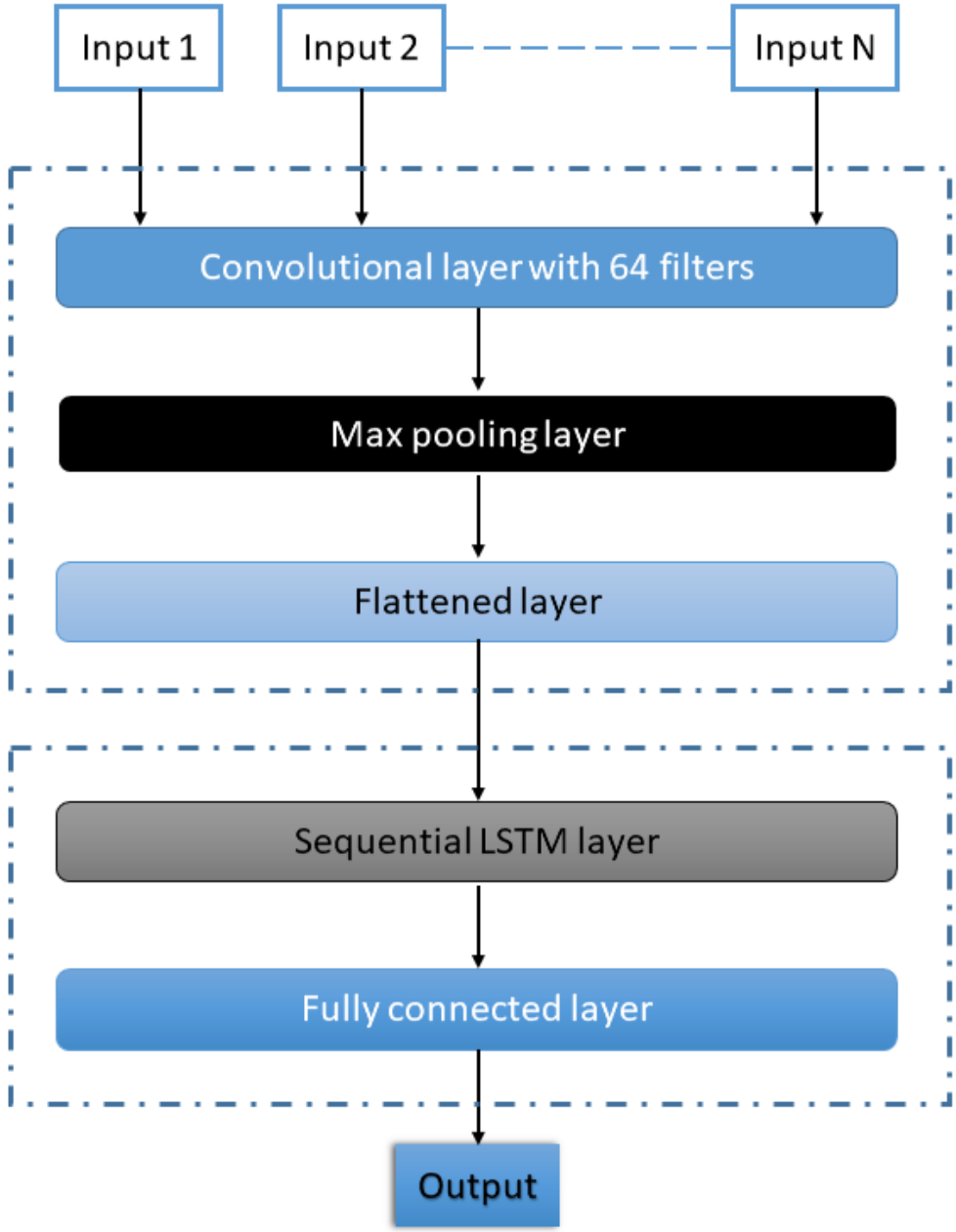


Figure 3.4. The applied hybrid CNN-LSTM model structure.

3.4. Evaluation Metrics

This thesis involves some applications that implement prediction and classification applications. Thus, the evaluation metrics range from various error rates to classification accuracy metrics. For instance, when working with regression models, the target variable can take on any real value, so it's highly unlikely that our model can accurately forecast the target variable. However, because we want to know how close our predicted values are to the ground-truth values, we apply a metric that measures total deviation (Pentreath, 2015).

Through surveying the literature, we realized that the most popular error metrics used for time-series forecasting applications are classified into two groups: scale-dependent and percentage-dependent metrics. Therefore, to assess the error between Y_i and the predicted value $\hat{y}_i$, the Mean Squared Error (MSE) and Root Mean Squared Error (RMSE), the Mean Absolute Error (MAE), the R-squared coefficient, and Mean Absolute Percentage Error (MAPE) are utilized (Al-Qaness et al., 2021). The mathematical formulae for each of these metrics are as follows:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \quad (3.1)$$

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i * 100}{y_i} \right| \quad (3.2)$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}} \quad (3.3)$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y}_i)^2} \quad (3.4)$$

Where y_i, $\hat{y}i$, $\bar{y}_i$, and n represent the observed value, the anticipated sensor value, the mean value, and the total number of observations, respectively. As scale-dependent metrics are appropriate to contrast different time-series data sets, and as part of our goal is to validate the model's skills on other data sets, we incorporate the MAPE error metric for this purpose (Hyndman, 2006).

On the other hand, when forecasting deals with categorical values, for example, YES and NO, the model's performance evaluation should include metrics such as accuracy, precision, and recall (Mehta, 2017).

1. **Accuracy**: One metric for evaluating classification models is accuracy. Formally, accuracy is defined as follows: Accuracy equals the number of correct predictions divided by the total number of predictions. For binary classification, such as in this instance, accuracy can also be calculated using positives and negatives.

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (3.5)$$

2. **Precision**: Precision in a classification task is calculated by dividing the number of true positives (correctly labeled items in the positive class) by the total number of items labeled as positive (the sum of true positives and false positives). Precision aims to address the following inquiry: What was the accuracy rate of positive identifications?

$$Precision = \frac{TP}{TP+FP} \quad (3.6)$$

3. **Recall**: It denotes the ratio of retrieved instances that are relevant. Recall refers to the measurement of the proportion of true positive cases that were correctly identified.

$$Recall = \frac{TP}{TP+FN} \quad (3.7)$$

Let TP represent True Positives, TN represent True Negatives, FP represent False Positives, and FN represent False Negatives.

3.4.1 Statistical significance test

The t-test is a statistical procedure utilized to ascertain whether a significant difference exists between the means of two groups. It aids analysts and researchers in determining whether the observed differences between the groups are statistically significant or merely coincidental. Although various t-tests exist, the Paired Samples T-test is the most prevalent. Comparing the means of two related groups, such as measurements obtained from the same subjects at distinct time intervals or under varying conditions, is the purpose of this method. One example is comparing the blood pressure of individuals before and after a treatment. The t-test is predicated on the t-statistic, computed by dividing the variance of the two groups' means by the standard error of the variance. A critical value derived from the t-distribution is subsequently applied to the result to ascertain whether the observed difference is statistically significant (T. K. Kim, 2015).

The t-test operates as follows:

- *The null hypothesis (H0)* posits that no significant difference exists between the two categories.
- *The alternative hypothesis (H1)* posits that a substantial distinction exists between the two categories.
- *The t-statistic* is computed by comparing the actual difference between the groups to the difference that would be expected under the null hypothesis.
- *Evaluation Against Critical Value:* The t-statistic that has been computed is assessed against a critical value derived from the t-distribution, typically set at a significance level of 0.05. The alternative hypothesis is accepted as the null hypothesis is supported by the calculated t-statistic exceeding the critical value. This indicates that there is a statistically significant difference between the groups.

4. TEMPERATURE AND WEATHER FORECASTING

4.1. Introduction

A plethora of diverse devices has been connected to the internet in recent years, creating a dynamic information-based ecosystem. Moreover, the abundant flow of this information is a key aspect of these future Information Technology (IT) ecosystems. This connectivity of the various kinds of objects, sensors, and end devices through the internet is called the Internet of Things (IoT) (Sniderman et al., n.d.). There are billions of end devices connected to the Internet and this number is rising sharply, causing a flood of data from a variety of sources (Maiti et al., 2018). IoT encompasses a wide range of applications such as smart cities, healthcare systems, agriculture, etc., and its increasing popularity poses various challenges like task management, decision-making, and stream processing, among others (Chegini et al., 2021).

Time-series data form a specific domain within IoT applications. It is a chronologically indexed series of data points, where each data deeply relies on the previous data (Y. Li et al., 2019). It has many applications that range from clinical medicine, human behavior prediction, traffic flow forecast to financial projection. That said, the complexity involved in time-series prediction, compared to traditional predictive tasks, requires us to pay more attention to the dependencies of the input variables if we intend to design and implement high-accuracy and trustworthy predictive models (J. Z. Kim, Lu, Nozari, Pappas, & Bassett, 2021). A good example of time-series data sets is weather forecasting.

The scientific method of predicting the state of the atmosphere based on certain time frames and locations is known as weather forecasting (Hayati & Mohebi, 2007). By solving a complex system of nonlinear mathematical equations based on specific mathematical models, Numerical Weather Prediction (NWP) uses computer algorithms to produce a forecast based on current weather conditions. These models define a coordinate system that splits the earth into a three-dimensional grid. Temperature, winds, solar radiation, water phase change, heat transfer, relative humidity, and surface hydrology are all measured within each grid, as well as their interactions with neighboring grids, to forecast future atmospheric attributes (Lynch, 2006).

With the growth of technology and computer science, meteorology embraced a more quantitative approach; and hence, prediction models became more accessible to researchers, forecasters, and other stakeholders. Many NWP systems, such as the Weather Research and Forecasting (WRF) model, have been developed in recent years as high-performance computing capacity has enabled the refinement and introduction of regional or limited-area models. As a result of its higher resolution rate, accuracy, open-source nature, community support, and a wide range of usability within multiple disciplines, the WRF model became the world's most used atmospheric NWP model (Hewage et al., 2021).

In data analysis, a regression equation that predicts the number of sunspots in a year was used to start the study of time-series prediction (Hewage et al., 2021). As a result, regression models have become the most fundamental and significant time-series prediction models. However, sophisticated models struggle to provide high-precision prediction due to the great complexity, irregularity, randomness, and non-linearity of actual data. Therefore, one can create nonlinear prediction models based on a vast quantity of historical data using machine learning approaches. We can obtain more accurate predictions than traditional statistical-based models through repeated training cycles and learning approximations (X. Yu et al., 2020).

In real applications, data changes over time due to implicit reasons, and hence, models trained on historical data become obsolete for this sort of evolving data. Consequently, detecting when this change occurs and updating the model accordingly are the main two challenges in practice (Žliobaitė, Pechenizkiy, & Gama, 2016). For instance, the types of this change/drift could be: (i) gradual over time; (ii) a recurring change; or (iii) an abrupt change. Regardless of the type of change, a common challenge in streaming data is that it is not stationary and machine learning models dealing with them must be flexible and adaptive. Hence, research has been encouraged to detect the drifts in streaming data systematically (Hackeling, 2017; Webb, Lee, Petitjean, & Goethals, 2017). Since meteorological data continuously arrives and evolves over time, the manual adjustment of the models is ineffective, and with the fast growth of the incoming data, it becomes infeasible as well. Therefore, the predictive models must be capable of updating automatically to deal with these problems. This is called adaptive learning (Mehta, 2017). Unfortunately, the use of such techniques has not been yet applied much in the smart city applications (Zliobaite et al., 2012).

To date, and to the best of our knowledge, there are no attempts to examine applying the notion of adaptive learning for streaming data on the collaborative fog-cloud architecture. Most of the work carried out on streaming data and adaptive learning ignores the possibility that we could leverage the high computation resources on the cloud and simultaneously use the proximity of fog nodes to the IoT devices for faster prediction and interaction. Hence, we aim in this thesis to propose a conceptual theoretical framework that increases the performance of the adaptive model by offloading the computation task of the model's construction to the cloud layer, while the model's usage is carried out on the fog layer. To elaborate, the proposed system collects the data streams from IoT devices to the fog nodes. Next, the data is forwarded to the centralized and global node (i.e., the cloud), where a global model is retrained and updated periodically. After that, it is generalized and allocated to the lower layer nodes (i.e., fog nodes), where the prediction of upcoming new data is performed locally. In this way, we could ensure a system model that provides high performance and low latency. The hypothesis that will be tested is that the collaboration of fog and cloud computing technologies should enhance the performance - speed-wise and accuracy-wise- of the adaptive learning model when is applied on streaming data for IoT applications.

Furthermore, to improve the performance of the temperature prediction model, we attempt to replace previously studied machine learning models with the state-of-the-art deep learning models for time-series data. In addition, we extend the proposed model to deal with both the univariate and multivariate problems. Also, we enhance the model by applying the adaptive learning concept so that the model is periodically updated using the most recent historical data whenever its prediction ability deteriorates. Consequently, the proposed approach in this study is designed as the following: In the beginning, three different models from the LSTM family are trained and evaluated on 80% as training and 10% as validation sets. The best model out of the three is chosen based on the lowest root mean square error (RMSE), among other metrics. Then, the concept of adaptivity (i.e., the model is adaptive) is applied to the three best-selected models to enhance their future forecasting. After that, the three models are reassessed using the last and the most recent 10% of the data set (testing set). To the best of our knowledge, this is the first time the bidirectional LSTM and adaptive learning are used on an actual data set (i.e., Istanbul Metropolitan Municipality meteorological data set) at a scale.

Thus, the proposed method of this study are as follows: The meteorological data is gathered and preprocessed according to the forecasting mode. i.e., whether the input is univariate or multivariate time-series, which means taking only the temperature as an input in the former or taking all the four features as an input in the latter. Thereafter, the pre-trained bidirectional LSTM model forecasts hourly the future temperatures (i.e., the output). Then, when the accuracy of the pre-trained model starts to deteriorate until it passes the specified threshold, the involved adaptive learning component, which updates the model fitting on the most recent historical data, intervenes to regain the desired accuracy and performance.

The experimental results illustrate that leveraging the concept of adaptive learning while applying deep learning to time-series data processing leads to a reduced prediction error rate. In other words, we were able to raise the accuracy by approximately 45% compared to the one achieved by the baseline model, i.e., relying only on shallow and constant learning.

This study has the following contributions:

- Initially, we investigated Apache Spark's adaptive learning method for streaming data (i.e., distributed and parallel processing).
- Secondly, we employ a distributed learning strategy by allocating the activities of creating and utilizing machine learning models to the cloud and fog computing layers, respectively.
- Then, we show how to use bidirectional LSTM and adaptive learning to make a good model for time-series predictions (temperature).
- After that, we use, test, and compare several cutting-edge deep learning models in meteorology, especially to predict temperatures. After that, we check them against well-known ML regression models.
- Next, we look closely at the various situations where either binary or multivariate forecasting is used.
- In the end, we showed that using adaptive learning with a single-variable time-series forecasting model gets better results with less memory and time.
- For forecasting purposes, we got a real set of meteorological data from the Istanbul Metropolitan Municipality site, extracted it, engineered it, and took samples from it.

4.2. Methods and Materials

This part provides a description of the proposed system model, an introduction to the model workflow, and an overview of the test scenarios.

4.2.1. The collaborative fog-cloud architecture

To deal with the problem of permanently change-prone data streams generated from IoT devices, this study suggests employing adaptive learning techniques on fog-cloud architecture to accomplish finer performance; that is, improving prediction and reducing latency. In the beginning, streaming data coming from different IoT devices (i.e., bottom layer), in our case different deployed sensors across Istanbul, is sent forward to the nearest fog node (i.e., middle layer). These fog nodes are distributed computing entities that are less computationally resourceful than cloud computing nodes, which could vary from routers and switches to computers and mobile phones. Then, the collected data on various fog nodes is dispatched to the cloud for global model training or updating. The training phase occurs when the model is constructed for the first time using all the historical data gathered from all those IoT devices through fog nodes as an interface. While the updating phase happens periodically, for example, daily, monthly, seasonally, etc.

After the training/updating process, the global model is distributed among the fog layer to all the nodes for local usage (i.e., prediction). In the end, these enhanced and periodically updated models are used for predicting future events, in our case the temperature for the next hours, days, or even months. This model's efficiency is based on two assumptions: (i) fog nodes do not have full data so they are insufficient to train the model individually as well as the limited resources of them could rise problems when processing data at scale; (ii) Fog nodes are closer to IoT devices, and hence, latency is low for prediction. Figure 4.1 depicts the proposed model's design.

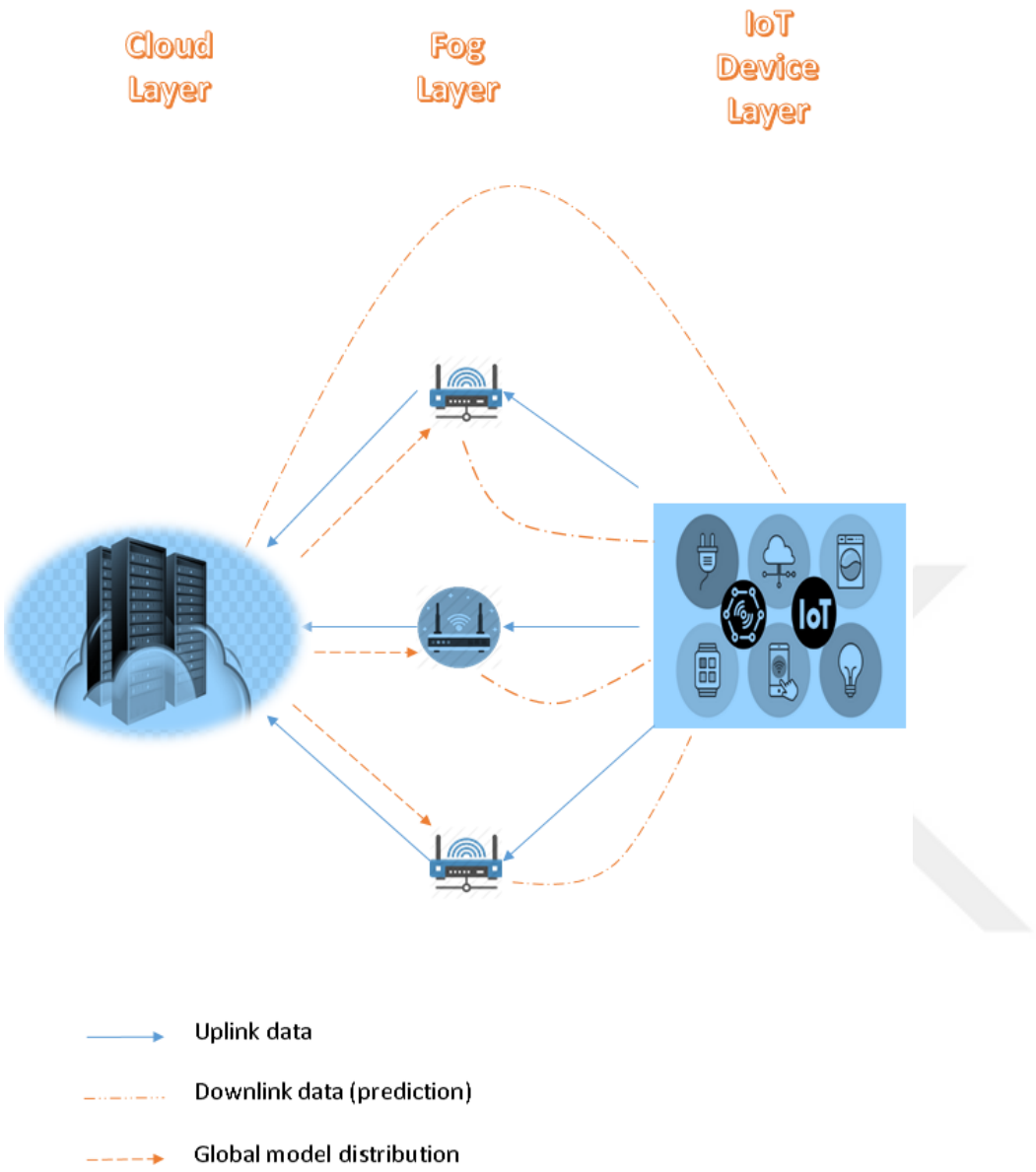


Figure 4.1. The proposed architecture of the fog-cloud system.

4.2.2. Data split and preparation

In this subsection, we describe the train, validation, and test harness applied in this work. We split the data sets into train and test sets with the percentages of 90% and 10%, respectively. The first years and months of the data sets (from January 2020 until March 2021) are taken for training purposes whereas the last 10% periods (from April to May 2021) are devoted to testing. As known, dealing with time-series data is different from static and not timely-related ones. Thus, the test set must be always from periods ahead of the training test and not vice versa. Thus, we rely on a specific type of data sampling which is called "expanding window splitter". It lengthens the training series over time, with each new

window preserving the whole history of the series up to that point. Although for each window the length of the testing series (forecast horizon) is fixed (24 hrs.), the step-back values range from 6 to 24 as will be shown in the experimentation section. The training set comprises 10,368 samples, whereas the testing set consists of 1,440 samples.

Before we fit the models to the data sets, we must transform the data into acceptable shapes for both the deep and adaptive learning styles. So, the below steps are performed prior to the fitting process of the model:

- We convert the time-series data to be stationary. We use a lag=1 difference to eliminate any trends in the data set.
- We transform the time-series data into a supervised learning problem. That is, read the observation(s) of the previous step(s) and have them as input for the current forecast step.

4.2.3. Proposed adaptive model pipeline

We employ a rolling-forecast scenario, often known as walk-forward model validation. The test sets time steps are walked one at a time. The actual predicted value from the test set is taken and made available to the model for the forecast in the following time step. This simulates a situation in which fresh temperature observations would be provided every hour and used to anticipate the upcoming hour or hours. Figure 4.2 simply demonstrates the proposed workflow for this study. As can be seen, this adaptive model is designed and evaluated to forecast the weather temperature (output). Nonetheless, the input is either the four weather features (temperature, humidity, wind, and precipitation) or only the temperature. i.e., a multivariate mode for the former and univariate mode for the latter. Thus, and as the temperature is the sole output for this model, its forecast values are compared with the actual ones for the feedback operation. Firstly, the mode of prediction must be selected when the input time-series data arrives. We may choose either univariate or multivariate predictions. Accordingly, the received input is filtered and preprocessed to accord with the input structure of the trained model. Then, it is fed to the LSTM model for future temperature forecasting. After that, and as a feedback mechanism, the predicted value is compared with the observed real one. If the accuracy of the model is sustained, then the process normally continues. Otherwise, the model is updated to cope with the seasonal changes.

The adaptivity adopted in this study works as follows: the forecasts are stored in a temporary buffer for a while. It could be a day, week, or month as the stakeholders wish. After the completion of the waiting phase, for example, a week, all the gathered forecasts are compared to the actual values, and an error score is computed to evaluate the effectiveness of the model. We employ the root mean squared error (RMSE) because it penalizes significant errors. In conclusion, if the error rate exceeds the specified threshold, it means there is a concept drift, and the model must be updated using the recent historical values.

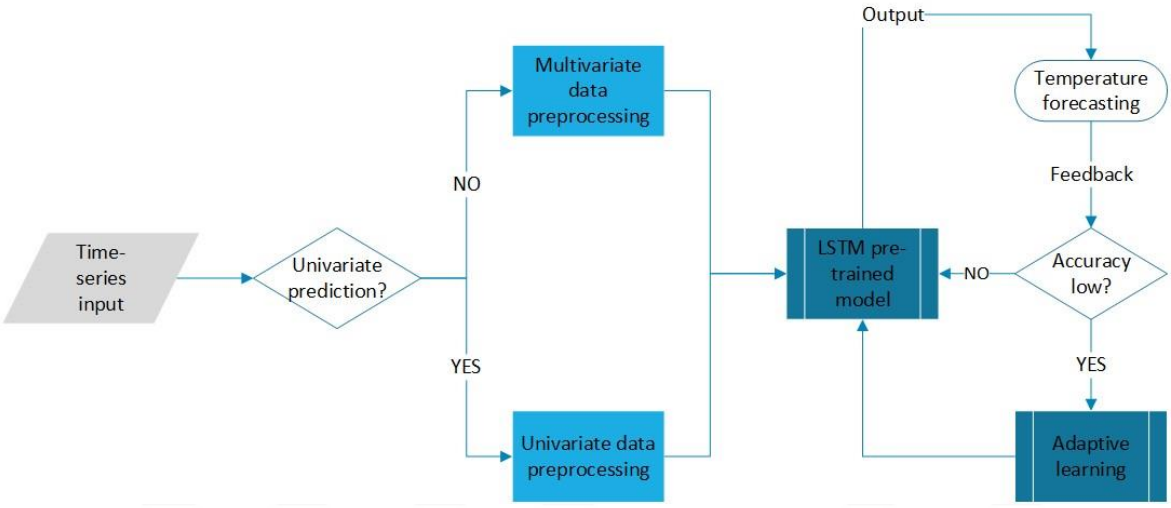


Figure 4.2. The proposed temperature forecasting adaptive model.

4.3. Results and Discussion

The following tools, software, and hardware were utilized to complete this investigation effectively. We utilize Keras to implement all the LSTM model versions on Python 3.7 using the Spyder IDE, which comes with Anaconda Navigator. All the tests were carried out on a computer with the following specifications: an Intel Core i7 8th generation CPU, 16 GB of RAM, and the Microsoft Windows 10 operating system, except for the tests executed on the Google Colab, which has different specifications that are described later in the text. Additionally, Apache Spark version 3.1.0, which is an analytics engine ("Apache Spark™ - Unified Engine for Large-Scale Data Analytics," n.d.), was used to achieve high performance for streaming data on both fog (by local device/computer) and cloud (by Databricks community) layers. Both MLlib and Streaming libraries provided by Spark were implemented via Python 3 API. Additionally, to execute the proposed model and algorithms, spark-submit tool was used.

4.3.1. Comparison of meteorological features

By and large, it is well-known that meteorological data sets compose of temperature, humidity, rainfall, wind speed and direction, and other features. So, before starting with our main experiments, we envisaged that we ought to focus on one output feature that would provide us with the best results. Accordingly, some empirical experiments were done on a subset of the gathered IMM data set, resulting in the conclusion that the temperature feature is the best candidate for the predicted output value. The experimental setups were as follows. Every feature has been experimented with 5 runs with and without applying adaptivity to the model. Besides, the 1-layered simple LSTM model was employed with the following hyper-parameters: 1 neuron, 1 lag, 500 epochs, and 2-update model. That is, the model is retrained and updated after every two new coming inputs.

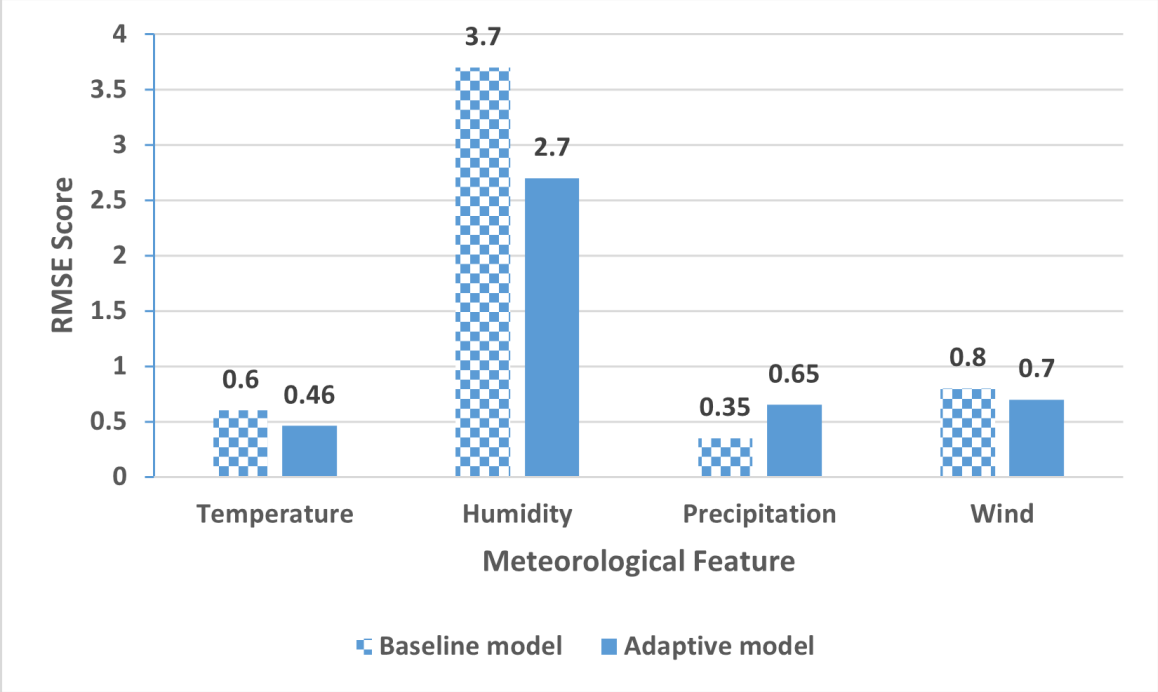


Figure 4.3. The RMSE quantifies the responsiveness of the most frequent meteorological characteristics to the adaptation process.

Figure 4.3 illustrates the various RMSE values obtained when conducting predictions on a feature basis. It is obvious that the precipitation, temperature, and wind achieved the lowest RMSE values, respectively. However, the humidity showed a great high value, which excluded it immediately from the selection process although these large RMSE values could be attributed to the larger weights used to measure the relative humidity. That is, the average

temperature, precipitation, wind, and humidity vary as 8.36, 0.20, 1.27, and 78.07, respectively. Thus, this interprets the huge difference in the RMSE results between the humidity and other features. As the differences between the two points in humidity are larger, the RMSE punishes it more. Moreover, during applying adaptive learning to the four features, we noticed that the temperature, humidity, and then wind features showed positive responses, for example, the temperature prediction was improved by about 30%, whereas the humidity and wind were enhanced by almost 25% and 5%, respectively.

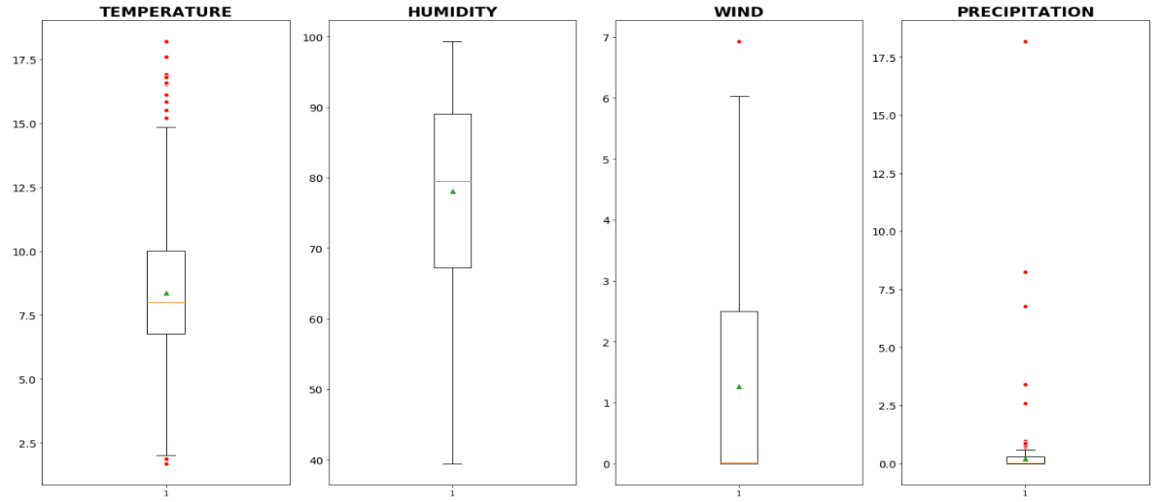


Figure 4.4. The analysis of the IMM four meteorological features using box plots.

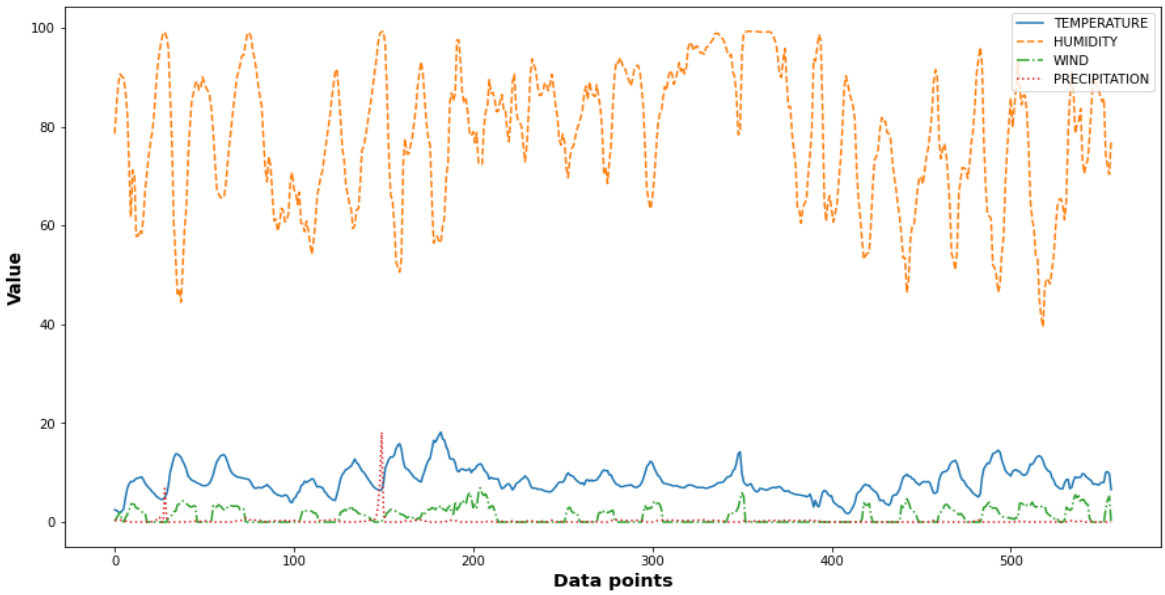


Figure 4.5. The trend diagram for the four meteorological features of the IMM.

Yet, the precipitation displayed the weakest response, where instead of improving the prediction results, the adaptive model makes them even worse and the RMSE value

increased from 0.325 to 0.665. One reason could be due to the minuscule values of this feature and mostly they are near zero, we think it affects the model skills and degrades its accuracy. To verify our drawn conclusions, we include the boxplot and trend chart displayed in Figure 4.4 and Figure 4.5, respectively. They confirm that the temperature feature is more well-distributed than the others, and hence, interpreting the above-mentioned conclusions and explanations. Consequently, and due to the popularity of temperature prediction in the literature and its manifold applications, it was picked for the rest of this study as an output feature for the prediction process.

4.3.2. Analysis of adaptive machine learning models

A few tests were done before we began implementing our deep learning model to determine the best ML prediction model for our application and data collection. We chose some of the most common regression models in ML, such as linear regression, Generalized Linear Regression (GLR), decision tree, Gradient-Boosted Tree Regression (GBTR), and random forest, because the challenge involves prediction. A portion of the overall data set was used to train and evaluate the five models. The average MAE, RMSE, and R-squared values from the preliminary trials of 5 runs are compared in Table 4.1. The random forest produced the best results, as shown in Table 4.1, and was thus chosen as a baseline for eventual comparison with our suggested model. There are 20 trees in the model, and the features' number is 3.

Table 4.1. A comparative analysis of ML prediction model.

ML models	**MAE**	**RMSE**	R^2
Linear regression	0.086	0.112	0.550
GLR	0.068	0.089	0.721
Decision tree	0.041	0.071	0.810
GBTR	0.033	0.059	0.845
Random forest	0.039	0.057	0.894

Following that, we employed a data set of about 12k cases, divided into 11k and 1k for model learning and testing, respectively. The idea is to forecast the temperature for the next hour based on the previous n hours' temperatures. As a result, each row was organized as follows: (Y, X_1, X_2, X_3, ..., X_n), where Y denotes the label whereas X_i indicates the features vector.

The goal of this experiment was to contrast the baseline and adaptive models on an exact test set while using different learning methodologies. For both models,

Table 4.2 displays the MAE, RMSE, and R-squared.

Table 4.2. A comparison of the performance of the baseline and adaptive random forest models.

Method	**MAE**	**RMSE**	R^2
Baseline model	0.769	1.077	0.971
Adaptive model	0.656	0.864	0.981

The average of the absolute variations between the expected and actual objectives is known as the MAE. It functions similarly to RMSE, with the exception that it does not penalize large deviations as severely. The model's prediction ability improves when the value for both decreases. As seen in

Table 4.2, the adaptive model outperformed the baseline model by a significant margin. It came close to achieving a 20% decrease in RMSE. Furthermore, the R-squared coefficient is a statistic measuring how successfully a model fits a data set, ranging from 0 to 1, with 1 representing a perfect fit. In

Table 4.2, both models provide impressive results, but the adaptive model has a little advantage. The t-test was conducted and proved that the progress achieved in the adaptive model is statistically significant where p-value was around 0.03.

4.3.3. Analysis of adaptive deep learning models

To elevate and improve the prediction accuracy, the second logical step was to implement a deep learning model(s) instead of the previously proposed machine learning ones. Thus, and according to the literature investigation conducted, we decided to tackle this problem by applying a Recurrent Neural Network (RNN) DL model, especially the LSTM model because it is famous for dealing competently with time-series data. Having that said, the LSTM model itself has several architectures and designs suggested in the literature earlier. Therefore, and after searching nearly all of them, we came to the decision that three of them are worth trying, which are stacked LSTM, bidirectional LSTM, and CNN LSTM.

As shown in Table 4.3, the lowest RMSE values were acquired from bidirectional LSTM, CNN LSTM, then stacked LSTM, respectively, where the bidirectional LSTM reduced the error prediction rate by almost 10% compared to the best adaptive ML model. Hence, and in terms of future trials and following subsections, bidirectional LSTM was exploited. Naturally, the weather forecasting data sets contain *n*-features, as said above, such as temperature, humidity, rainfall, and wind speed. Moreover, deep learning algorithms can perform not only single-input-single-output (univariate) predictions but also multi-input-single-output (multivariate) ones. So, we experimented with the robustness of the LSTM models on both prediction modes using different time steps (i.e., look-ahead).

Table 4.3. An evaluation of the three LSTM models that were proposed, with a focus on the effects of adaptive learning and variable quantity on prediction errors (RMSE).

LSTM Models	**Multivariate baseline**	**Univariate baseline**	**Univariate adaptive**
Stacked LSTM	0.913	0.810	0.770
Bidirectional LSTM	0.910	0.740	0.593
CNN LSTM	0.952	0.780	0.750

The results proved that all LSTM models accomplished better prediction accuracy when the univariate model is implemented. Also, we observed that the more the lookbacks were, the higher the RMSE value was. That is, for multivariate mode, it is better to stick with fewer time-steps. On the contrary, for univariate predictions, the more the time-steps are, the less the value of RMSE is. Table 4.3 illustrates the variation between RMSE values of both univariate and multivariate different LSTM models. Again, it is worth noticing that the prediction error rate is lessened by approximately 10% or more.

Most importantly, by obtaining better results with univariate prediction, we can benefit from the truth that fewer input features mean less time computation and memory consumption when the model is trained and constructed. Especially when the DL models are well-known for their high-complexity training cost. Hence, by applying the univariate model on the bidirectional LSTM model, not only can we get higher accuracy, but also, we save power and memory consumption. Since our past experiments revealed that using adaptive learning on ML models always surpassed the performance of the baseline in terms of accuracy, we carried out the same comparison to measure the validity of adaptive learning on DL models as well. To this end, we employed the 11k samples included in the IMM data set and trained the model on them frequently; in other words, applying adaptive learning. Likewise, the

model is constructed and used to predict temperature using just a univariate model, for the sake of simplicity. Finally, the time-steps applied in this experiment were 6, 12, 18, and 24.

Table 4.4. The adaptive bidirectional LSTM model forecasts temperature for a variety of callbacks.

	RMSE	
Time-steps	**Training**	**Testing**
6	0.67	0.61
12	0.75	0.64
18	0.63	0.56
24	0.60	0.56

We can infer from the table the following: (i) The training and testing RMSE values are almost similar, which means there was no over-fitting or under-fitting issues occurred during the training process; (ii) The more the time-steps we set, the less prediction error we get. All in all, as it is depicted by Table 4.3, the adaptive model displays higher accuracy than the baseline one, and it decreases the whole error prediction rate by approximately 20%.

Table 4.5. A comparison between the temperature forecasting capabilities of our proposed model and those of cutting-edge LSTM-based models.

Proposed model	**RMSE**
MISO-LSTM Hewage et al. (2021) (Hewage et al., 2021)	1.669
TCN LSTM Hewage et al. (2020) (Hewage et al., 2020)	1.431
The proposed model in this study	0.593

Compared with the recent and state-of-the-art studies, which concern temperature forecasting and implementing LSTM-based models, We can infer from the table the following: (i) The training and testing RMSE values are almost similar, which means there was no over-fitting or under-fitting issues occurred during the training process; (ii) The more the time-steps we set, the less prediction error we get. All in all, as it is depicted by Table 4.3, the adaptive model displays higher accuracy than the baseline one, and it decreases the whole error prediction rate by approximately 20%.

Table 4.5 reflects the RMSE scores gained by different proposed methods. Our optimal adaptive bidirectional model surpasses the recently proposed models that are relevant to this study by more than 50%. Having that said, it is significant to point out that the data sets used in these three studies are not the same although they are approximately have the similar timespan for training and testing, which is less than two years. Yet, attaining the same benchmark and comparing the different proposed models accordingly will have more

conclusive insights and fairer comparisons. This is why the upcoming subsection will discuss the efficacy of the proposed model on other data sets and compare our outcomes with what is published in the literature using the same data sets. Also, it is worth noting that t-test was used to verify the performance of all the experiments and the null hypothesis was always rejected.

4.3.4. Adaptive model validation using different data sets

After proving the robustness of the adaptive bidirectional LSTM model on the IMM data set, we designed an experiment consisting of 4 trials with different settings for the time-steps, hyper-parameters, and the callbacks during the training process. Furthermore, we employed two other meteorological data sets, which are widely used in many previous studies. The aim was to validate the proposed model for temperature forecasting by applying it on different data sets to evaluate whether the adaptive model will still lead to a better prediction accuracy? Hence, the summary of these experiments is briefly presented in Figure 4.6. As expected, the adaptive bidirectional LSTM model always shows fewer prediction errors regarding the RMSE. That said, it may be worth stating that the model reaches the lowest RMSE values in different time-steps for the three data sets. By this, we mean that the best time steps settings for both the IMM and KMA were either 18 or 24, whereas it was between 12 and 18 for the Beijing PM2.5. The reason could be attributed to the following: the more the data points are non-stationary and fluctuated, the closer we should look back (i.e., fewer time-steps back). In other words, it is obvious that the Beijing PM2.5 data set has more volatile temperature values than the other implemented data sets. This means that different data sets, even if they are in the same domain, always need human intervention to tweak the model settings and hyper-parameters for the best results.

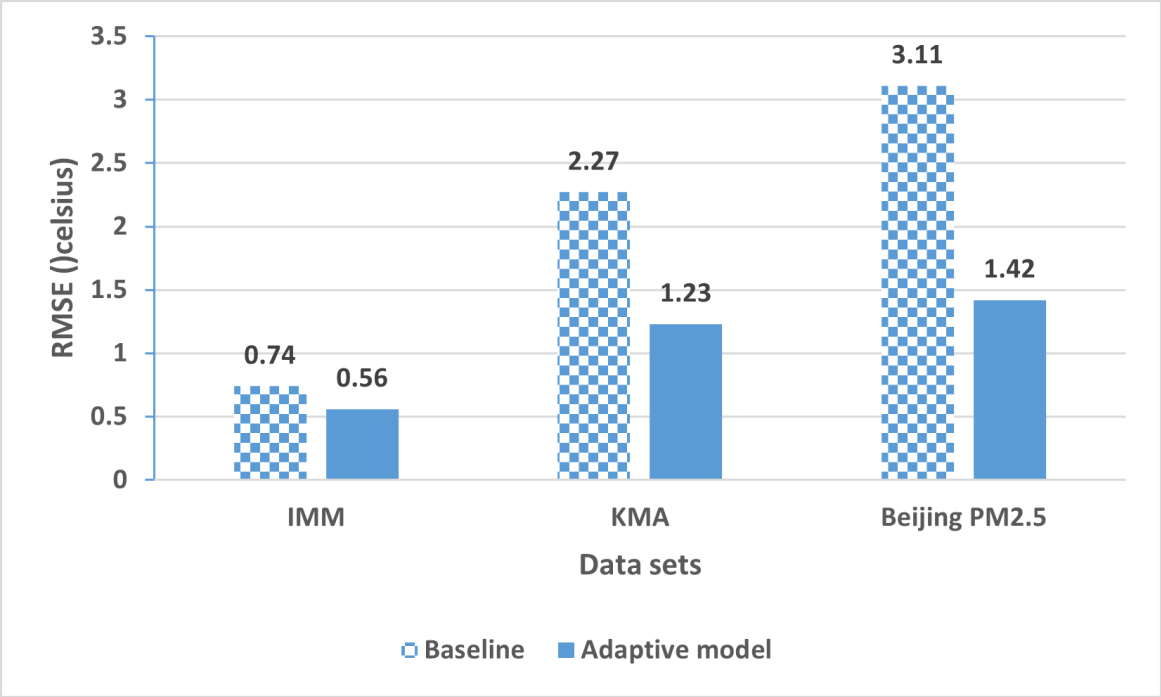


Figure 4.6. When validating the robustness of the model, the RMSE of the baseline and adaptive LSTM is compared across three distinct data sets.

Another observation is the variance among the RMSE values of the three data sets. The KMA and Beijing PM2.5 exhibit rather large RMSE values, about double compared to IMM's. This could be attributed to the oldness, poor gathering, and inconsistency of the data set though we had pre-processed them before they were used. Yet, to confirm the model's goodness among the three data sets, we add a new error metric, which is MAPE. It is a popular metric amongst data scientists to calibrate forecasting models for different use cases and data sets, making it easier to understand for end-users and simpler to interpret for researchers. As shown from Table 4.6, the experimental outcomes of MAPE are 7\%, 8\%, and 10\% for IMM, KMA, and Beijing PM2.5, respectively, showing that the model's performance is between very good and good. When comparing them with the baseline model values, we can see that the proposed model always achieves the lower MAPE, and thus, provides more accurate temperature forecasting. Accordingly, we could state that the designed model in this study works well on temperature forecasting regardless of the use case or data set meant to be applied.

Overall, the application of adaptive learning in time-series data processing has proven promising and more suitable for such ever-changing data. Although the random forest algorithm outperformed its peers, we were able to reduce the prediction error by

implementing a special Recurrent Neural Network (RNN) called LSTM. In addition to applying LSTM to the multivariate problem, various data sets in the same field were used to compare them with the benchmark. Table 4.7 shows how the prediction error rate is reduced by using different algorithms and models.

Table 4.6. MAPE-based comparison of the baseline and adaptive models for the three data sets.

	MAPE	
Data set	**Baseline LSTM**	**Adaptive Bidirectional LSTM**
IMM	9%	7%
KMA	12%	8%
Beijing PM2.5	13%	10%

Table 4.7. The entire development of this research.

Algorithm	Model	RMSE	Comments
Random forest	Baseline	1.077	No adaptations or data updates have been made.
Random forest	Adaptive	0.846	The prediction error is reduced by almost 20%.
Bidirectional LSTM	Baseline	0.740	By applying the bidirectional LSTM, the prediction rate continued to decline.
Bidirectional LSTM	Adaptive	0.593	A remarkable improvement in forecast accuracy has been achieved. However, model building can be quite computationally expensive.

Our main goal is to achieve high performance and low latency, as these are essential requirements for real-time or near-real-time data processing in IoT applications. In this context, we propose the idea of leveraging end-device/fog and cloud collaborative architecture so that the model's building time is reduced to accomplish low latency. The following subsection discusses the benefits of such architecture by demonstrating the time differences in model construction between a standalone computer and a cloud server. Finally, experimental findings have confirmed that the proposed adaptive deep learning model reduces the prediction error by almost 45% compared to the baseline model. However, deep learning models are computationally exhausting and require high maintenance. Therefore, the trade-off of high accuracy against high computation power must always be considered when selecting the appropriate model for our task.

4.3.5. Local vs. Server computation

In this experiment, we attempted to evaluate the robustness of the model on different computation powers and resources. To begin with, we tested the time elapses for both model training and data prediction. Thereafter, we inspected the scalability of the model since it is

designed to deal with big streaming data. Two main test scenarios are plotted: (i) doing the experiments on both a local device (a personal computer) and cloud computing server (Google Colab), (ii) increasing the number of epochs and data size during training to examine the scalability of the models on both platforms. The specifications used for the cloud computing server are displayed in Table 4.8.

Table 4.8. The specifications of the exploited cloud server.

Parameters	Google Colab
GPU	Nvidia K80
GPU Memory	12GB
GPU Memory Clock	0.82GHz
Performance	4.1 TFLOPS
No. CPU Cores	2
Available RAM	12GB
Disk Space	358GB

In comparison, as shown in Table 4.9, and for both data sets, the prediction errors on either the local device or the server almost remain the same, which indicates the robustness of the model no matter where it is deployed. Secondly, the closeness of the training and testing prediction scores guarantees that the model has not been affected by any overfitting or underfitting problems.

Table 4.9. Using a total of ten epochs to compare the efficacy of adaptive models on a local device and the Colab server.

	Local device				Server			
	Training time	Prediction time	Train score	Test score	Training time	Prediction time	Train score	Test score
IMM	193.33 s	3.33 s	1.20	1.16	27.94 s	2.28 s	1.20	1.21
PM 2.50	221.11 s	3.60 s	2.15	2.40	29.33 s	2.26 s	2.19	2.35

In contrast, it is worth noting that the training time on the server (the time unit is in seconds), using high computation power and ample memory resources, is lessened up to 85.5% for both data sets, especially when the GPU processing is implemented. For studies, such as (Hewage et al., 2020, 2021), which argue that stacked LSTM is lighter weight and more efficient than bidirectional LSTM, although we agree with their experimental outcomes, we prove that shifting the training process of the model to a higher computation layer, for instance, cloud or fog computing, results in less training time. In addition to the efficiency, we gained here, we accomplish better forecasting results when choosing bidirectional LSTM over lightweight one. Furthermore, the more epochs are used, the higher accuracy we obtain. From Table 4.10, it can also be seen that the forecasting scores, measured by RMSE, gradually decrease, leading to the significance of scalable models that can operate at a large

scale of data inputs and train deeply by a modest number of epochs. Yet, at some point, we must stop increasing the number of epochs, especially when they achieve the same results with fewer ones. From Table 4.10, we can notice that almost similar scores are accomplished at 50 and 100 epochs though they have massively different times for model construction. As a result, 50 epochs are adequate for this study's requirements and goals.

Table 4.10. Analyzing the influence of epoch counts on the efficacy of the model.

Colab using GPUs	Training time	Forecasting time	Training score	Testing score
10 epochs	27.94 s	2.28 s	1.2	1.21
50 epochs	97.98 s	2.52 s	0.67	0.55
100 epochs	148.53 s	2.70 s	0.63	0.57

Overall, from this sub-experiment, we conclude that training the model on computationally powerful cloud servers always results in less training time, higher accuracy, and better scalability. This inference proves the correctness of our last study hypothesis and motivates us to move on to federated learning for not only distributing the workload between the servers and end devices but also for assuring more end-user privacy.

4.4. Conclusions

This study argued that merging adaptive learning with a bidirectional LSTM model for time-series forecasting could improve the performance of the forecasting compared to conventional algorithms. Moreover, it proved that the migration from ML algorithms to DL ones, especially by using LSTM different model types, enhances the overall prediction process, where the RMSE decreases from 0.846 to 0.593. Although LSTM models might require significant computational resources, i.e., time and memory, this could be solved by implementing distributed processing. The study set out to examine which forecasting model could achieve satisfactory results, is it ML or DL models? Accordingly, we attempted to enhance the best-selected algorithms by consolidating adaptive learning to them. Then, we applied this enhanced model to both univariate and multivariate predictions to assess the overall accuracy of prediction and the capability of operating at scale. Finally, to validate the model's robustness and goodness, two other meteorological data sets were employed.

The results of this study indicate that the combination of bidirectional LSTM with an adaptive learning concept is the best solution for forecasting the temperature and updating

the model periodically produces better forecast results. The ultimate adaptive model reduces the prediction error rate by nearly 45%, and the time required for building the model could be reduced to approximately 85% when cloud computing is applied. Also, when the model is applied to different time-series meteorological data sets, the MAPE is always between 8-10%, which indicates that the proposed model is very good and vigorous. Having that said, we plan to promote these models from just adaptive learning to federated learning to handle both the scalability and privacy concerns. In other words, data analysis will be carried out by distributed machine and deep learning methods. Moreover, we intend to move on and apply the designed model to another area of smart city applications; however, the focus will be on streaming time-series forecasting applications such as flood or traffic forecasting.

5. FLOOD FORECASTING

5.1. Introduction

The issue of flood forecasting (FF) has gained significant attention in recent times due to the impact of climate change. Across the world, the changing climate has resulted in less predictable flood patterns. To effectively manage floods, it is crucial to have precise information on their occurrence, extent, and impact on the environment (Hingmire & Bhaladhare, 2022). The ability to predict floods in coastal cities is crucial for authorities to make well-informed and effective decisions when managing flood activities. This helps to ensure the best outcomes in terms of efficiency and safety for all involved (Dai et al., 2021). Thus, flood forecasting is an essential solution to save lives and protect infrastructure during natural disasters. Yet, while hydraulic models are dependable in predicting floods, they take a considerable amount of time to do so. Hence, the need for machine learning models based on supervised learning has arisen (Z. Zhang et al., 2021).

The Internet of Things (IoT) is extensively utilized in flood monitoring, forecasting, and management. It is an integral component of a smart flood forecasting system, providing predictions and alerts to the public regarding potential floods (Piadeh et al., 2022). Smart sensors that relate to hydrological information technology have been installed in various urban areas. These sensors monitor changes in flood water levels. Examples of these sensors include depth meters, hydrological monitoring cameras, intelligent water level gauges, and combinations of various sensors like hydro-meteorological sensor nets (Moishin et al., 2021) and automatic weather stations (Piadeh et al., 2022), which collect vast amounts of data. The rapid development of data-driven hydrological forecasting, including machine learning (ML), has resulted from the recent increase in flood-related data, the creation of intelligent algorithms, and the increased application in hydrology (Naik, Verma, Patil, & Hingmire, 2021), deep learning (DL) (Chen et al., 2021), and computing infrastructure (Birch et al., 2021).

As the inundation process during a flood is highly complex and uncertain- especially under non-stationary conditions, accurate flood forecasting remains a challenge (Q. Zhang et al., 2017; Z. Zhang et al., 2021). For instance, the lack and imbalance of the research data sets are one of the biggest problems faced during the data collection and model training

processes. Besides, some raw flood data is missing due to sensors misreading or damage during heavy precipitation, and accordingly, creative and effective preprocessing steps should be implemented and advised. Last but not least, the insufficient existing performance of the data-driven models such as machine and deep learning models drives the need for continuous investigation in this context to provide efficient, robust, and accurate prediction models.

This study investigates the application of weakly supervised adaptive deep learning to project floods in coastal, IoT-enabled, smart cities. According to our best knowledge, weakly supervised adaptive learning has not been utilized before in this domain of smart cities. Specifically, the following are the contributions of this study:

- As a preliminary step, weakly supervised learning (WSL) is utilized for labeling and augmenting data to provide effective imputation techniques to fill in missing data. It is crucial to have accurate data for real-time flood prediction models.
- Then, a specialized type of recurrent neural network (RNN) known as bidirectional long-short-term memory (LSTM) is applied to address the problem of time-series prediction and classification appropriately.
- Adaptive learning is subsequently applied to the designed model to address the seasonality issue with such rapid data.
- As computational time and prediction range should be emphasized more in future research, it is suggested that distributed architectures be utilized for learning. For instance, cloud-fog collaborative architecture and federated learning (FL). The purpose of such a proposal is to provide sufficient lead time for operators to make preventative decisions.

5.2. Methods and Materials

Based on current understanding, there is a limited body of literature that focuses on leveraging unlabeled data for the purpose of improving FF. This indicates a potentially fruitful area for further research. Moreover, it has been observed that a significant number of studies have focused on addressing the issue of utilizing unlabeled data in feedforward neural networks (FF) by employing either supervised learning (SL) or semi-supervised learning (SSL) methods. In addition to the requirement of possessing a properly annotated

subset for employing SSL, the application of WSL techniques in FF to address the challenge of unannotated data is evidently constrained and necessitates thorough investigation.

After collecting meteorological data, primarily precipitation, the process of annotation begins. Annotation in this context involves categorizing the sample as either having experienced water inundation (flood) or not; In other words, working with a binary class. Yet, the process of annotating big data is laborious, costly, and time-consuming. Consequently, it is essential to switch from manual to automatic (programmatically labeled) annotation. In this context, this work suggests employing weakly supervised learning (WSL) methods for automated data labeling, aiming to optimize time and cost efficiency. WSL can be utilized for either data labeling or data augmentation.

Data labeling addresses the problem of unlabeled data sets in the following manner: First, several labeling functions must be defined to use them to project the class label in the future without additional human intervention. To avoid fallacies and pitfalls during the automatic annotation of the big data, it is recommended that these functions be designed and written by domain experts. The reason for this is that the less precise and reliable a model (classifier/predictor) is, the noisier the labels are. Second, these functions are applied to the training and validation sets to evaluate the performance of labeling. When the level of accuracy is unsatisfactory, the labeling function set is rewritten or expanded until a level of acceptable accuracy is achieved. Thirdly, a labeling consensus is reached using a label model supplied by the Snorkel API package. The combined data set and new labels are then stored in a database or warehouse. Lastly, the preprocessed data set is utilized for model development (training and evaluation). Figure 5.1 displays how labels are generated from labeling functions using weak supervision techniques.

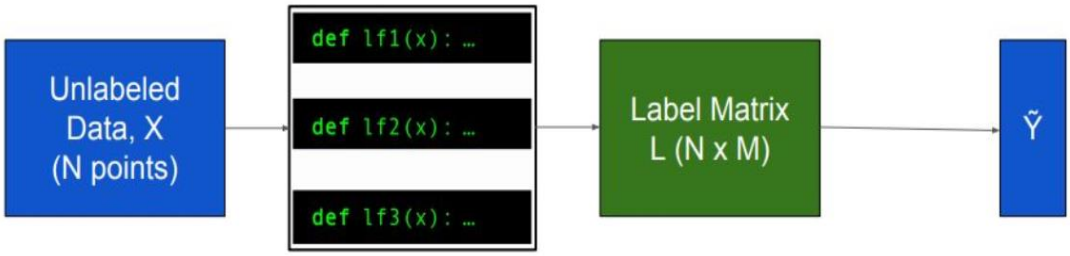


Figure 5.1 Labels generation using WSL.

In contrast, data augmentation necessitates a small number of well-labeled data sets to produce additional examples. Simply put, transformation functions are created and used to generate examples with identical labels. Increasing the size of the data set could significantly enhance the performance and precision of the model by exposing it to larger examples and greater diversity. Therefore, the trained model becomes more robust and reliable for classification or prediction in the future. Figure 5.2 depicts the preprocessing steps utilized in this study. As shown in the figure, both data labeling and augmentation are employed to address the issues of laborious data labeling and a lack of sufficient samples for model construction.

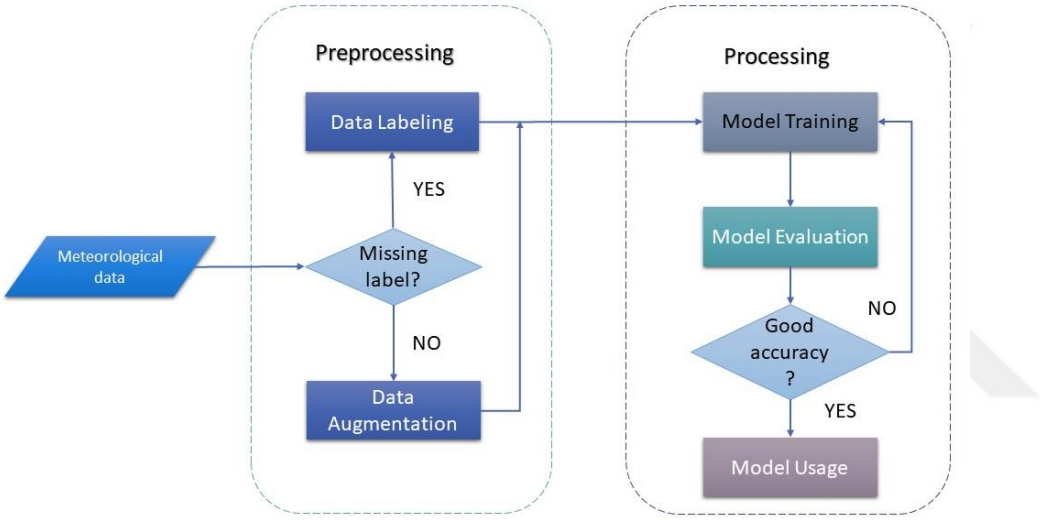


Figure 5.2. The flowchart of weakly supervised learning preprocessing methods.

Figure 5.3 depicts the proposed workflow for this investigation. The selection of the forecasting mode is contingent upon the arrival of time-series input data, such as rainfall. There is a choice between univariate and multivariate predictions. The input is filtered and preprocessed according to the model's input structure. The data is subsequently inputted into the LSTM model for future flood prediction. The predicted value is compared to the observed value as a feedback mechanism. Accordingly, if the model's accuracy is maintained, the process typically continues. Otherwise, the model is revised to accommodate seasonal shifts.

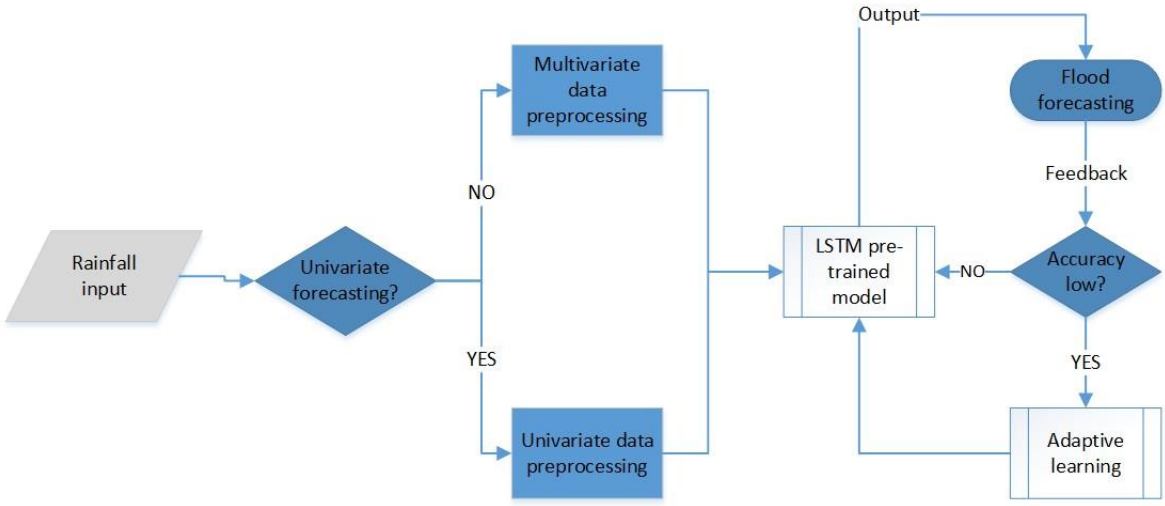


Figure 5.3. The proposed flood forecasting model.

5.2.1. Experimentation setup

The tools and libraries used in this research are Python packages, including Pandas, NumPy, and Sklearn. Firstly, they are used to upload, preprocess, and convert data sets into a suitable format for subsequent processing. Then, for WSL, the Snorkel tool is used to weakly label and augment unlabeled instances. After that, Python libraries such as Keras and TensorFlow are employed for deep supervision learning. Finally, the experimentation is conducted as the below scenarios:

- Baseline model: the bidirectional LSTM model is trained and evaluated using a subset of data points with ground-truth labels.
- Weakly Supervised Learning: it is applied as a preprocessing step:
  a. *Data Labeling*: Generate programmatically labels for unlabeled data using various labeling functions and model labeling techniques such as MajorityLabelVoter and LabelModel, then train and evaluate the above model on them.
  b. *Data Augmentation*: Besides labeling functions, this work applies data augmentation using Snorkel to increase the size of the train set, then it trains and evaluates the model on them.
- Adaptive learning: it is implemented on all the previous scenarios to boost the performance of forecasting models.

- Ablation study: An ablation study is conducted to assess the performance of an AI system by selectively removing components to determine the individual contribution of each component to the overall system. So,
    a. Applying only data labeling.
    b. Applying only data augmentation.

5.3. Results and Discussion

In this section, the obtained results are represented, discussed, and concluded with insightful ideas and future guidelines.

5.3.1. Labeling functions (LF)

A snippet from the labeling functions code is given in Figure 5.4. Many labeling functions have been defined to calculate the appropriate label for the flood class. In other words, each function automatically decides whether the class label is true or false, where true means the probability of flood occurrence is high whereas false means no flood is going to happen. Writing solid functions requires domain expertise and decent historical data comprehension and investigation. Although the experience of the authors is extremely modest in the field of flood risk and management, we succeeded in acquiring a good understanding of the data set so that the results' accuracy of the written labeling functions is acceptable. Arguably, regarding the data set size and dimensions, it does not need that much knowledge and expertise to work it out.

```
from snorkel.labeling import labeling_function

@labeling_function()
def is_flood_ann(x):
    if x.RAINFALL >= 2900:
        return FLOOD
    else:
        return ABSTAIN

@labeling_function()
def is_no_flood_ann(x):
    if x.RAINFALL < 2900:
        return NO_FLOOD
    else:
        return ABSTAIN

@labeling_function()
def is_flood_season(x):
    SUM = x.JUN + x.JUL + x.AUG + x.SEP
    if SUM >= 2100:
        return FLOOD
    else:
        return ABSTAIN
```

Figure 5.4. A segment of the labeling functions code.

5.3.2. Analyzing the labeling performance

Before applying the labeling functions on big unlabeled data, they must be tested on a subset of a well-labeled data set. So, we analyze their performance through several statistical measures like coverage, conflicts, and most importantly, empirical accuracy. If the accuracy is good enough and acceptable, then the functions are applied to large-scale unlabeled data. After designing and applying the labeling functions on these unlabeled sets, we receive their labels and combine them, and store them into a storage unit like a CSV file, for example. From Figure 5.5, we can notice that the accuracy of two-thirds is 100% whereas it is 90% for one-third, which is relatively adequate accuracy.

	j	Polarity	Coverage	Overlaps	Conflicts	Correct	Incorrect	Emp. Acc.
is_flood_ann	0	[1]	0.583333	0.375000	0.000000	14	0	1.0
is_no_flood_ann	1	[0]	0.416667	0.041667	0.041667	10	0	1.0
is_flood_season	2	[1]	0.416667	0.416667	0.041667	9	1	0.9

Figure 5.5. Labeling functions' performance analysis.

5.3.3. Evaluating the labeling functions

In weakly supervised learning for labeling functions, researchers evaluate the labeling function using Snorkel-supported labeling models to reach a consensus among the labeling functions. This step is crucial because selecting the correct labeling model affects the final output label's approach and precision. Thus, this work tested two well-known labeling models: (i) LabelModel, a model for learning the LF accuracies and combining their output labels; (ii) MajorityLabelVoter, a model for determining the majority vote on a label. As can be seen in Table 5.1, the LabelModel provides greater precision, recall, and accuracy. Consequently, it was chosen as the labeling model for this study.

Table 5.1. The performance of the labeling models is subjected to analysis.

Labeling Consensus Model	Accuracy	Precision	Recall
LabelModel	100%	100%	100%
MajorityLabelVoter	95.8%	90%	100%

5.3.4. The results of the four previously designed scenarios

The below bar chart, Figure 5.6, displays the accuracy results for all predefined scenarios in the experimentation subsection, which includes and investigates the impact of applying weakly supervised learning as a preprocessing technique and adaptive learning as a complementary algorithm to the time series deep learning models. Overall, it is seen that the accuracy gradually improves from 58.33\% to reach the best score, which is 81.31%, when both weakly supervised and adaptive learning is implemented.

Regarding the preprocessing techniques used, utilizing labeling functions increases the model's accuracy from 58.33% to 63.84%, while merely applying data augmentation techniques boosts it to 69%. Yet, when the two techniques supported by weakly supervised learning are implemented together, the improvement of the model’s accuracy could reach approximately 73%. The lower accuracy obtained by the labeling functions is that the unlabeled data in hand is not large enough to boost the model’s accuracy, especially when it is known that deep learning models must have big data to forecast and classify sufficiently. On the other hand, Data augmentation techniques can significantly increase the size of the

data by applying appropriate transformation functions, designed, and implemented by subject matter experts.

Similarly, applying adaptive learning to either baseline or other improved models enhances the total performance, achieving an ultimate accuracy of almost 81\%. Undeniably, the consideration of adaptive learning to time series data points, especially seasonal ones, is crucial to detect the degradation of the model during long periods. Figure 5.6 confirms this theory as every time the adaptive model is implemented, the accuracy goes up. In conclusion, utilizing weakly supervised learning techniques for data preprocessing and then augmenting the processing model with concepts of adaptive learning could improve the accuracy of flood forecasting models. It is important to mention that a t-test is performed to determine whether the differences observed between different scenarios and models are statistically significant. As the p-value consistently came out to be less than 0.5, the null hypothesis is rejected, indicating that the improvement is not due to random reasons or samples, but rather, is statistically significant.

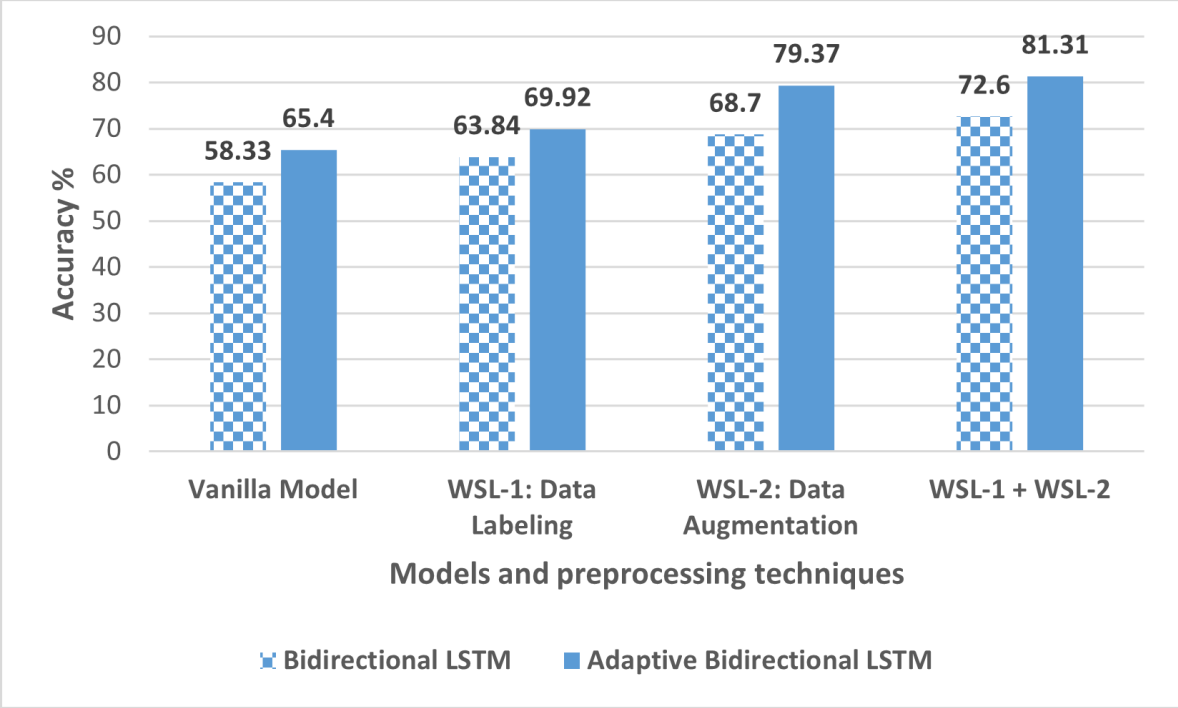


Figure 5.6. The accuracy of the proposed model and its enhancement steps.

Nonetheless, flood forecasting using LSTM with only one input characteristic (rainfall) is considered limited and inadequate. There are numerous explanations and justifications for our selection. Firstly, rainfall is frequently the primary cause of flooding occurrences, and it

has a considerable impact on river levels. In many circumstances, particularly in short-term forecasting, rainfall alone can provide useful information for anticipating future floods. Second, data availability and quality are important factors. Rainfall data is often more widely available and well measured than other variables. Using a solid and abundant data source is critical for developing a strong forecasting model. Third, utilizing a single input component, like rainfall, simplifies the model while simultaneously improving interpretability. This can be useful for stakeholders who do not have a thorough understanding of complex model architecture. Fourth, more complicated models with various input features may require additional computational resources and training time. Using rainfall as the only input feature helps speed up the model construction process. Furthermore, our model is intended for short-term forecasting when the immediate impact of rainfall is more significant. In short-term circumstances, additional factors such as land cover changes or river morphology may have a minimal impact when compared to the immediate rainfall conditions. However, we recognize the potential advantages of including additional characteristics like river flow data, soil moisture, or topography. Yet, we emphasize that the decision to begin with rainfall as the sole input feature was purposeful, and that future versions of the model could investigate the inclusion of additional variables for incremental improvement. As seen in the visual above, the model performs well in terms of reliability and accuracy, supporting the efficacy of the chosen approach.

5.4. Conclusions

This study focuses on the issue of flood prediction by exclusively utilizing meteorological data, with a specific emphasis on the rainfall variable. In order to address this matter, our initial step was accessing a publicly available dataset pertaining to flood prediction, which was derived from the probabilities of monthly and annual rainfall. Subsequently, as a result of the limited scale and presence of noisy or incomplete labels within these data sets, the aforementioned data set undergoes preprocessing via weakly supervised learning methodologies. Two techniques, namely labeling and transformation functions, are employed in this study. The labeling function is utilized to assign labels to missing classes, while the transformation function is used to increase the size of the data set. The primary objective is to enhance the performance of the forecast model. Therefore, the utilization of preprocessing techniques may contribute to the improvement of accuracy in deep learning models. This is because these models heavily rely on accurate and substantial data samples.

Based on current understanding, these techniques are considered groundbreaking in the field of machine learning and were initially employed in the aforementioned study and domain.

Once the data points were gathered, preprocessed, and stored, we implemented a robust deep learning model known as bidirectional LSTM. This model has been demonstrated to be highly effective for predicting time-series data. Furthermore, it demonstrated its efficacy, particularly in the domain of meteorological prediction. However, we advocate for the implementation of adaptive learning in the designed model to get the highest possible level of accuracy. Hence, the precision of the suggested model, incorporating the methodologies, concepts, and algorithms, exhibits an enhancement of around 40% in comparison to the baseline model.

In our forthcoming research, we intend to obtain a data set from one of the published studies pertaining to flood forecasts. This will enable us to undertake a comparative analysis between our proposed methodology and the approach employed in the study. Furthermore, the study will incorporate and analyze novel functionalities and data resources pertaining to flood calamities. Subsequently, as a coherent progression within the scope of this thesis, we aim to advance the existing model from conventional deep learning to the paradigm of federated learning. In essence, the utilization of distributed machine learning techniques will facilitate the provision of data analysis. Ultimately, there is potential for the model to be extended to additional domains within the realm of smart city applications. However, the primary emphasis will be placed on the analysis of time series streaming datasets.

6. HOUSEHOLD ENERGY CONSUMPTION FORECASTING

6.1. Introduction

Increasing demand and scarce energy resources require smart city applications to be able to predict how much power an entire city or a single building will need. To accomplish this, we must first predict the short-term energy consumption (load) of a single household in a building and then the building's total energy consumption (Peng et al., 2019). Consequently, it is possible to aggregate and forecast the demand for smart cities (A. R. Khan, Mahmood, Safdar, Khan, & Khan, 2016). The importance of short-term residential energy consumption forecasting stems from the decentralization of renewable energy generation, such as photovoltaic systems, which allows households to generate their own energy and maximize its use to become increasingly self-sufficient (Fallah, Deo, Shojafar, Conti, & Shamshirband, 2018).

Many studies in the literature have been conducted to forecast the energy consumption of households using artificial intelligence (AI) techniques and approaches (Yildiz, Bilbao, Dore, & Sproul, 2017). For accurate model training, however, the majority of AI models require vast quantities of historical data, which must be collected and stored in a centralized architecture (P. Zhang, Wu, Wang, & Bi, 2015). Now, the Advanced Metering Infrastructure (AMI) facilitates this process. Installation of smart meters in customer homes is key to the AMI, which allows for reliable data collection on load consumption at intervals of no less than 15 minutes (Asghar, Dán, Miorandi, & Chlamtac, 2017).

Privacy issues have been raised by regulatory agencies and others due to technological advancements that have introduced new choices for end-users and energy providers (Fan et al., 2010). The main theme of these concerns revolves around the utilization of user data for the purpose of uncovering client behavior. In certain regions, privacy concerns may lead consumers to decline the usage of smart meters (Mármol, Sorge, Ugus, & Pérez, 2012). Numerous privacy-preserving technologies employ techniques such as data aggregation and encryption to mitigate privacy concerns. These solutions aim to find a balance between safeguarding user privacy and enabling energy suppliers to access data. The integration of privacy-preserving approaches with residential short-term load forecasting systems gives rise to a paradox. The latter employs granular household data, a type of data that is not

compatible with privacy-preserving aggregated or masked data formats. The present state of AI-driven solutions results in an escalation in computational overhead for model training (Asghar et al., 2017). The significant processing intensity poses a limitation when it comes to expanding the system to accommodate millions of smart meters, a need that is steadily growing. In order to alleviate computational limitations, it is possible to train AI models on smaller portions of the data. However, this approach diminishes their ability to generalize well across the entire data set.

Edge computing is a paradigm that involves the decentralization of computational processes, hence bringing them in closer proximity to end users. The integration of cloud computing at the network edge enhances various aspects such as speed, efficiency, dependability, privacy, security, and scalability. It is ideal for energy consumption forecasting due to its privacy and scalability requirements (W. Z. Khan, Ahmed, Hakak, Yaqoob, & Ahmed, 2019). Edge computing, like other architectural models, exhibits certain limitations. AI and machine learning models achieve optimal convergence when all training data from different users are processed centrally. However, edge computing models train locally to ensure client confidentiality, reducing their ability to generalize (Peng et al., 2019).

On the other hand, in recent years, federated learning (FL) has garnered significant interest among researchers due to its capacity to integrate diverse data sources and facilitate the collaborative and privacy-preserving construction of machine learning models (B. McMahan, Moore, Ramage, Hampson, & y Arcas, 2017). Thus, FL enables multiple parties to build a model collaboratively without disclosing data (Kang, Liu, & Liang, 2022). Additionally, it refers to the machine learning strategy that enables decentralized model training across massively separate data sources without privacy concerns (X. Liu, Zhu, Xia, Jiang, & Yang, 2021). These private data sets of many dispersed clients and institutions come from a variety of sources, including smartphones and IoT devices (Diao, Ding, & Tarokh, 2021). Furthermore, the ability to generalize of both machine learning and deep learning methods is frequently contingent on training on diverse data sets, such as data from multiple medical institutions, which is a restrictive requirement given the sensitive nature of medical data (Kassem et al., 2022). Additionally, the implementation of this approach has demonstrated success in various application fields (Q. Yang, Liu, Chen, & Tong, 2019), including human-computer interaction (T. Yang et al., 2018), language modeling (Wu, Liang, & Wang, 2020), transportation (Pokhrel & Choi, 2020), and Industry 4.0 (Cioffi,

Travaglioni, Piscitelli, Petrillo, & De Felice, 2020), where privacy and/or scalability are essential.

In many complex data analysis scenarios, data undergoes temporal changes and necessitates near real-time analysis. The patterns and relationships in dynamic data often change over time, rendering models used for analysis obsolete. In the field of machine learning and data mining, this occurrence is referred to as “concept drift”. Certain domains, such as time series forecasting and predictions on streaming data, involve the ordering of predictions by time. In these domains, it is important to explicitly test for and address the problem of concept drift, which is more likely to occur. One common challenge in data stream mining is the lack of strict stationarity, where the underlying distribution of incoming data unpredictably changes over time. There is a growing need to promptly identify concept drifts in data streams (Brownlee, 2018).

In this paper, we employ adaptive federated learning techniques within a distributed architecture for the purpose of short-term energy load forecasting. In this framework, smart meter data is stored at the edge and utilized by multiple individuals for the purpose of training a global model. The model enhances customers' predictive capabilities by revealing unfamiliar local patterns observed by other users when it is returned to them. The forecasting model employed is LSTM, which leverages federated learning to uphold user privacy. Adaptive learning was incorporated into the overall architecture to enhance the accuracy and performance of the model. This was achieved by retraining the model whenever its accuracy falls below a predetermined threshold.

We evaluate our method carefully and show that it can make forecasts that are as good as, or even better than, an AI-based strategy that requires data centralization. In terms of how long it takes to train a model, the proposed framework is better than the centralized architecture, making it a more scalable and privacy-friendly alternative. Also, our strategy will reduce communication costs even more when energy consumption measurements are taken more often than every hour (i.e., with a smaller interval).

The novel contribution of this study consists of the following facets:

- A cutting-edge deep learning model is developed, employing federated learning methodologies, with the aim of enhancing the precision of energy consumption (load)

prediction within smart grid systems. This reduces communication overhead and privacy concerns in energy consumption and demand applications.

- Adaptive learning is employed to enhance forecast accuracy by updating the model in response to changes in the data.
- To evaluate the efficacy of the suggested methodology in comparison to the centralized architecture, a real-world data set is employed, and extensive experiments are conducted.
- Ultimately, we proceed with the design, testing, and evaluation of two discrete input-output typologies for the forecasting models. The designs discussed in this context consist of two main approaches: the univariate method and the multivariate approach. The univariate approach focuses primarily on energy use as the input variable. On the other hand, the multivariate approach combines extra data, such as calendar and weather information, with energy consumption.

6.2. Methods and Materials

Within this section, we shall describe the recommended model and distributed architecture, while also furnishing comprehensive information regarding the experimental design.

6.2.1. Proposed adaptive model

In our earlier studies for temperature and flood forecasting, we designed the general adaptive learning workflow depicted in Figure 6.1. This adaptive approach is applied in the current study to predict energy consumption for the next hour (short-term forecasting). The adaptability employed in this study functions in the following manner: Predictions are temporarily stored in a buffer. The duration of the project may vary, ranging from a single day to a week or even a month, contingent upon the preferences of the stakeholders involved. Following a designated waiting period, such as one week, a comparative analysis is conducted between all projected forecasts and the corresponding actual values. Subsequently, an error score is calculated to assess the efficacy of the model's performance. The use of root mean square error (RMSE) is justified due to its ability to penalize substantial errors. Finally, if the error rate surpasses the predetermined threshold, it signifies the occurrence of concept drift, necessitating an update to the model with the most recent historical data.

For example, the RMSE of the forecast from the previous day is compared to the average RMSE over a two-week period. If the RMSE observed yesterday is 10% greater than the average biweekly RMSE, the model undergoes retraining using the most up-to-date historical data. Otherwise, there is no adaptive training. The primary innovation of this study lies in the utilization of the suggested adaptive learning approach during the implementation of federated learning to forecast household energy consumption. This ensures that our model can effectively identify incidences of data drift while minimizing the associated expenses related to data transfer and retraining.

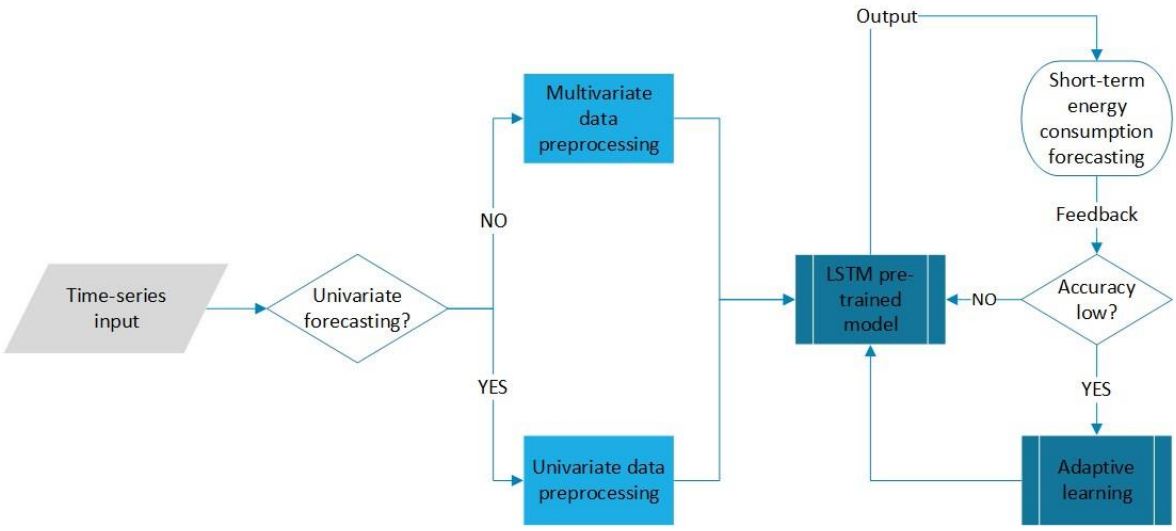


Figure 6.1. The proposed adaptive approach to forecasting household energy consumption.

6.2.2. Federated architecture

Federated learning (FL) is a decentralized approach to training machine learning models, wherein remote devices utilize their own data for local training. A centralized server aggregates the models into a global model, which is subsequently distributed to remote devices for additional training. The iterative nature of this process facilitates the inclusion of a substantial quantity of devices, enabling their participation without necessitating the transfer of data to a centralized location. FL demonstrates efficacy in handling non-independent and identically distributed (non-iid) data and presents advantages in terms of data exchange when compared to centralized training.

FL training workflow involves round-slotted time and two entities: end devices and a centralized server, with steps outlined in Figure 6.2. First, the process of initializing a

machine-learning model on a centralized server involves the random selection of weights. Additionally, the server transmits the model type and hyper-parameters to the end devices. Second, the training phase encompasses selecting a subset of end devices, transmitting the global model to these devices, conducting local training on edge devices using their respective data, and then aggregating all the received local models. Subsequently, the server generates a revised global model. The selection of end devices is reliant on the specific application at hand. Additionally, a commonly employed technique in this context is aggregating data on a centralized server, referred to as Federated Averaging (FedAVG) (Bonawitz et al., 2019).

FedAVG is an extension of FedSGD, wherein the process entails the utilization of Stochastic Gradient Descent (SGD) by end devices, followed by the transmission of model weights to the server (B. McMahan et al., 2017). This approach hinders the convergence of global model training by performing a single SGD step per round. The limitation mentioned is addressed by the FedAVG approach, which introduces the concepts of local epochs and batches. This approach aims to minimize the interaction between the server and devices by reducing the frequency of communication. Additionally, the weights are averaged in a manner that considers the number of training samples used locally.

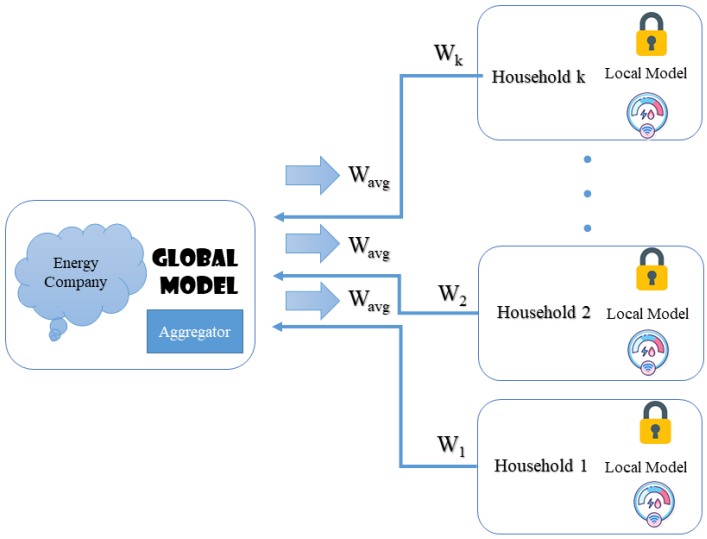


Figure 6.2. Federated architecture for forecasting domestic energy consumption.

To optimize the performance of FedAVG, it is crucial to fine-tune three parameters (Savi & Olivadese, 2021):

- *Nround*: The total number of rounds to be executed in the experiment.
- *Nepoch*: The number of local epochs that will be executed by each selected remote device in a single round.
- *Bsize*: The quantity of local samples utilized for training in each round or epoch, commonly known as the batch size.

In this study, we use a federated short-term energy consumption forecasting architecture illustrated in Figure 6.2, like that in (Savi & Olivadese, 2021). By collaboratively training an LSTM network, this architecture enables accurate energy consumption forecasting for residential properties. Model aggregation and redistribution steps are used to train a global LSTM model, as prescribed by the FL approach. This is because many edge devices train local LSTM networks by using locally generated energy consumption measurements. This architecture considers two entities:

- *Energy company*: It provides electricity to customers and, through smart grids, could increase the supply capacity.
- *Residential customers*: They consume the electricity supplied and are billed for by the energy company in their homes.

Three functional nodes are included and interact; their definitions are derived from previously defined federated learning functional nodes:

1) *Smart Meters*: All user premises have them installed, and the energy company is their owner. A smart meter logs energy consumption measurements (samples) with fine granularity (up to one per minute) (Amasyali & El-Gohary, 2018).
2) *Edge Computing Node*: It might be a GPU board or PC that the energy provider owns and installs at the customer's location. It is linked to the smart meter via a suitable wireless or wired network connection and is responsible for storing the smart meter's energy consumption measurements. It is also connected to the aggregator for model exchange and can interact with customers' end devices (such as a smartphone) via special-purpose apps used for data visualization and as an alarm endpoint for the edge computing node.
3) *Aggregator*: The energy company owns this central server which uses encrypted connections to communicate with the edge computing nodes. After each round, it

aggregates local models trained by multiple edge computing nodes. This operation produces a global model, which is subsequently redistributed to edge computing nodes.

The significance of this study is rooted in the introduction of adaptive deep learning into the conventional federated learning architecture. The purpose of this approach is to provide benefits in terms of data exchange when compared to centralized training, while also ensuring privacy. The global model is updated only when there is data drift, resulting in more accurate models with reduced data transmission and communication. Consequently, this approach leads to a decrease in forecasting error and training time. Our approach distinguishes itself from existing methods in academic literature on energy consumption by demonstrating a resilient and swift reaction to the drift concept. Previous studies (Savi & Olivadese, 2021; Taïk & Cherkaoui, 2020) have employed a basic federated architecture to predict energy consumption. However, these approaches have been unable to effectively account for the frequent occurrence of data drift throughout the forecasting process. Consequently, this resulted in a trade-off between the accuracy of deep learning models and the potential burden imposed on the network due to the vast amount of data required for repetitive training. In contrast, we have introduced the adaptive learning idea as a fundamental element inside the federated learning architecture, aiming to ensure the freshness (being up to date) of current models while minimizing the need for extensive data transmission between local devices and the central node.

6.2.3. Experiment setup

To evaluate the proposed model within the federated architecture, we use the parameters from Table 6.1. To implement and test the LSTM model, we employed the TensorFlow Federated framework[1] and the Keras API[2], and Google Colab Cloud[3] to perform the experiments.

As stated in Table 6.1, the training phase comprises the initial eleven months of 2013. The final month is then utilized as a test set. The rationale behind using only the most recent month is that the test set in time series must never be temporally observed by the model

[1] https://www.tensorflow.org/federated/
[2] https://keras.io
[3] https://colab.research.google.com

during the training phase. In addition, the number of lags fluctuates during experimentation to determine which lags produce the greatest results. Instead, the prediction horizon is always the following hour (short-term forecasting). In the simulation, the server and client settings are identical. In other words, the training and aggregation stages share the same parameters and functions. However, we altered the number of clients (participants) to test the influence of scalability on the system architecture, i.e., its impact on forecasting accuracy and complexity.

Table 6.1. Configurations for model testing, training, and simulation.

Parameters	Applied value/function
Training period	11 months
Testing period	1 month (December)
Time steps (lags)	6, 12, 24, 48 hours
Forecast horizon	next hour
Forecast interval	hourly
Number of clients	2, 4, 6, 8, 10, 20, up to 50

The provided data consists of the energy consumption recorded over a specific period: the previous *x* hours. This data is in univariate mode, meaning it only includes information about energy consumption and does not consider any other variables. Rolling-forecast validation, which is also referred to as walk-forward model validation, is utilized. The time steps of the test set are iterated sequentially. The model utilizes the extracted actual predicted value from the test set in the subsequent time step to facilitate forecasting. This scenario involves the simulation of hourly measurements of actual energy consumption, which are subsequently used to forecast the energy consumption for the upcoming hour or hours.

Consequently, due to the exclusive focus on energy consumption as the model's output, the feedback operation involves a comparison between projected values and real-world observations. Initially, the input undergoes a process of filtration and preprocessing to adhere to the prescribed input structure of the trained model. The data is subsequently fed into the Long Short-Term Memory (LSTM) model to forecast forthcoming energy loads. In order to provide feedback, the predicted value is subsequently compared to the actual observed value. If the precision of the model is upheld, the process will proceed in its regular manner. Furthermore, the model takes into consideration seasonal variations.

Since the target variable acquires a real number value while dealing with regression models, it is extremely unlikely that our model can exactly forecast the target variable. As a result,

we employ a measure that considers the overall deviation in order to determine the distance between our predicted values and the actual values (Pentreath, 2015). Several loss functions, such as the mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), and many others, are used to measure the error between y and the forecast value ŷ.

6.3. Results and Discussion

In this section, we describe our experimental findings and discuss the positive and the negative aspects. In addition, we contrast our proposed framework with state-of-the-art algorithms.

6.3.1. Exploratory data analysis (EDA)

In this study, univariate energy consumption is implemented. In other words, the only input to our model is the energy consumption of prior back steps (lags). Thus, we investigate and explore features of interest based on other temporal parameters. The primary characteristics of the chosen data set are displayed in Table 3.4. In addition, Table 6.2 illustrates the most popular energy consumption statistics to comprehend its behavior and implicitly understand subsequent outcomes from our proposed framework.

To gain a general understanding of the nature of the data set, we randomly selected one participant (household) and analyzed its energy consumption patterns, searching for trends, seasonality, and normal distribution. For example, Figure 6.3 depicts the energy consumption ranges from 2012 to 2014. The line graph reveals that there are seasonal peaks that exceed 2.0 kWh on occasion. Even though the maximum energy consumption for this data set is approximately 6 kWh, this household appears to be a moderate consumer.

Table 6.2. The 2013 energy consumption statistics for all participating households.

Statistic measure	**Value (kWh)**
mean	0.239
std	0.381
min	0
25%	0.062
50%	0.131
75%	0.256
max	6.528

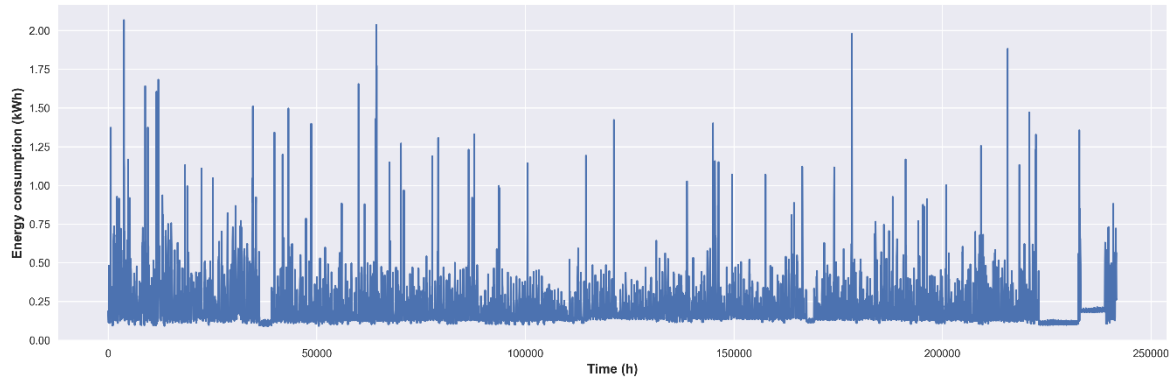


Figure 6.3. The hourly energy consumption of a representative residence.

Furthermore, Figure 6.4 depicts the histogram of the same household, from which we can infer that the energy consumption is right skewed. Most of the hourly energy consumption falls within the range of 0 to 0.25 kWh. The remainder of the energy load ranges between 0.5 and 0.7 kWh, with the remaining values being exceptional or outliers.

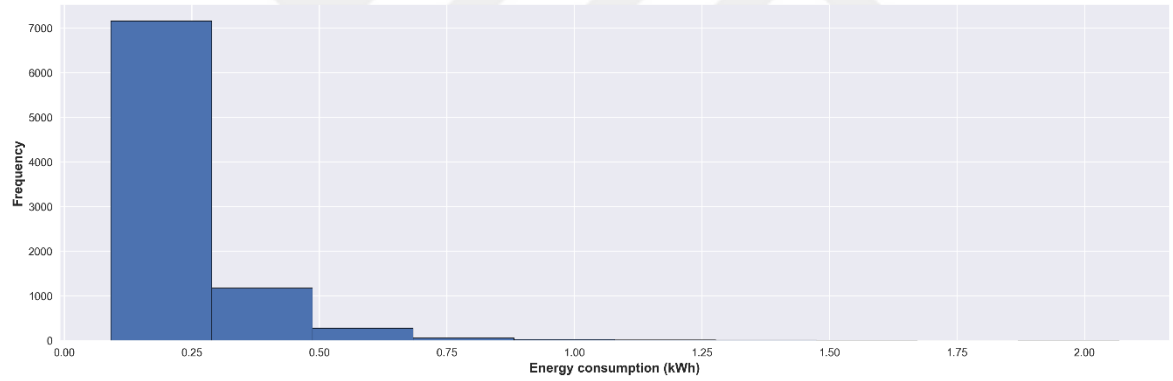


Figure 6.4. The energy consumption histogram of an arbitrary household.

To find out how far we should look back (lags), we resampled the data set (down-sample) to daily rather than hourly. Moreover, we calculated the autocorrelation and drew its chart to conclude how many lags could be beneficial during our experimentation. As seen from Figure 6.5, up to day 24 the lags could be effective. Yet, looking back up to day 4 (i.e., 4 * 24 = 96 hours) would be optimal.

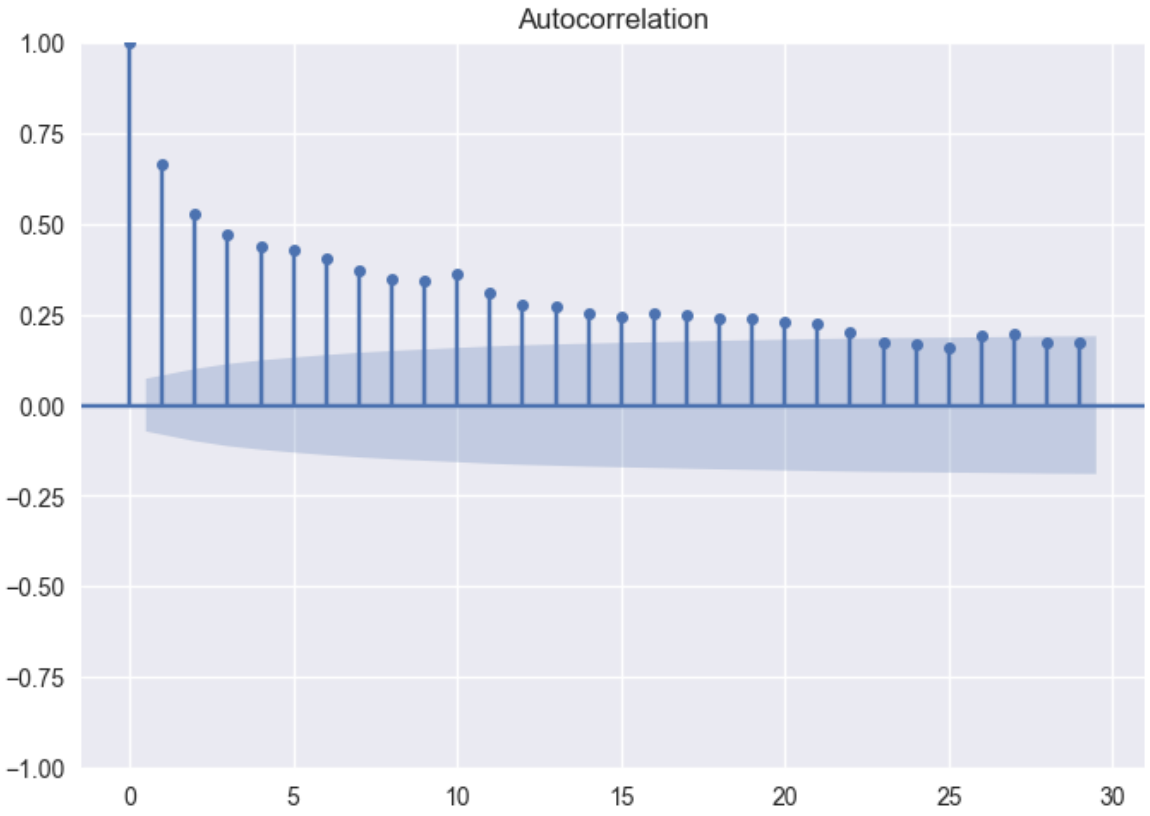


Figure 6.5. The lags in the autocorrelation of daily energy usage.

Finally, Figure 6.6 shows the observation, trends, seasonality, and residuals of the selected household in the mentioned period. As seen, there are some increasing and decreasing trends throughout the two years, but it is obvious that the trends of consumption soar in the fall and winter when the weather is colder while they decrease in the spring and summer. Also, from the seasonality chart, there is monthly seasonality. This leads us to investigate in the future if there is weekly and daily seasonality as well because it is going to help us understand the data better, and hence, design a better solution for it to maximize our objective.

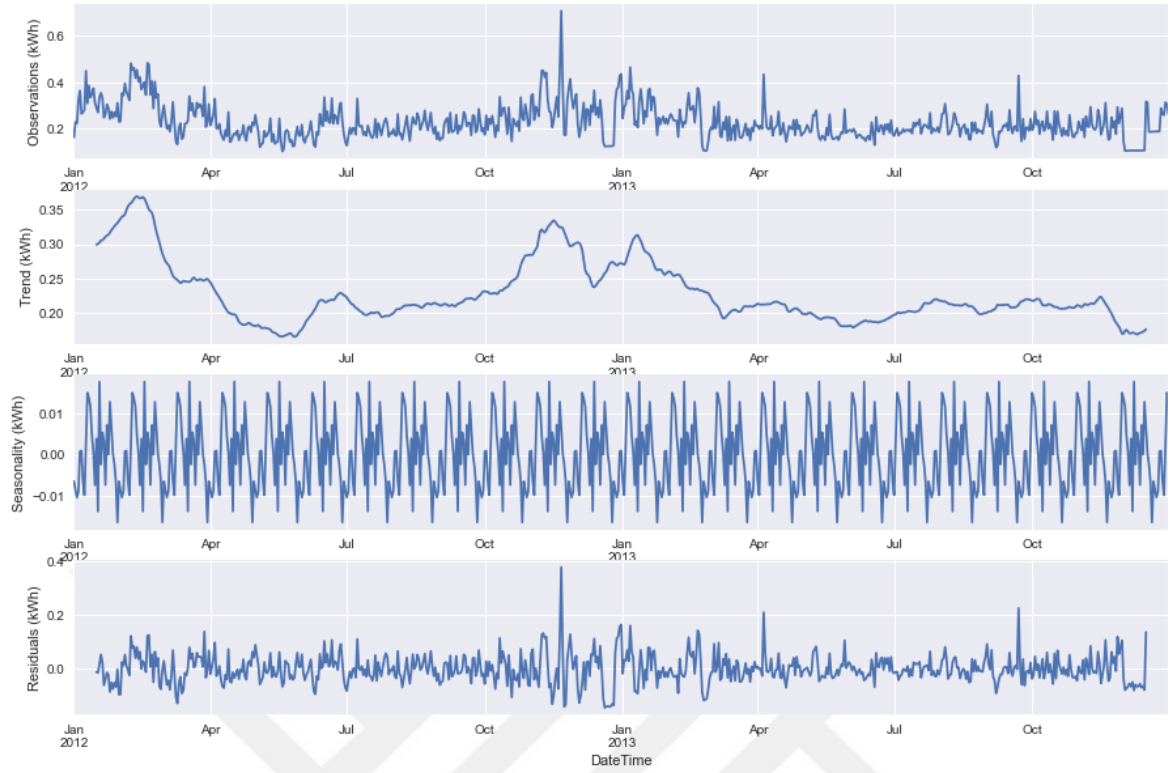


Figure 6.6. This figure examines the observations, trends, seasonality, and residuals of energy consumption in London households.

6.3.2. Impact of adaptive learning

In this part, we investigate the influence of adaptivity on the performance of models, specifically, and architecture, generally. As shown in Table 6.3, implementing adaptive learning with the deep federated learning architecture improved the system's performance by approximately 4% and 7% for MAE and RMSE, respectively. One could argue that the percentage of improvement is insignificant compared to the extra computational power, memory usage, and communication bandwidth needed for the retraining process when data drift is detected, and the model must react to it. There are two explanations for the lower improvement that adaptive learning produces in this specific experiment. First, the limited evaluation period (December 2013) prevented the occurrence of data drift as household energy consumption remains relatively consistent during the winter season. Therefore, it is unnecessary to update the model using our proposed algorithm. This resulted in a marginal improvement in the performance of the model. Furthermore, we expect that the effectiveness of adaptability will be demonstrated as the testing period is extended to encompass additional months and diverse seasons. Thus, adaptive learning may be less useful for time-series data if there is no occurrence of concept drift or data drift. However, these drifts are

common in time-series applications, so we argue that adaptive learning is worthwhile in this context. It is worth mentioning that the t-test is performed to determine whether the model performance improvement is statistically significant, rather than due to chance. The p-value was 0.02, which disproves the null hypothesis and confirms the alternative hypothesis.

Table 6.3. Comparing adaptive and non-adaptive learning regarding households' energy consumption.

Model	MAE	RMSE
Non-adaptive stacked LSTM	0.0892	0.1294
Adaptive stacked LSTM	0.0859	0.1208

6.3.3. Sensitivity to *N*-clients

We attempted to investigate the performance of the model by increasing the number of participants in the federated learning environment and examining the scalability of the architecture. As shown in Figure 6.7, as the number of clients increased, the MAE fluctuated. However, the performance remained stable after 20 or more participants. It appears that the model was exposed to enough examples for its performance to be steady. Furthermore, the number of clients in this study was limited to 50 for easier comparison with (Savi & Olivadese, 2021). However, we plan to repeat the experiment with a larger number of participants in a future study, as was done in (Alonso et al., 2020). Having said that, in all other experiments, we adhered to *n-clients* = 50.

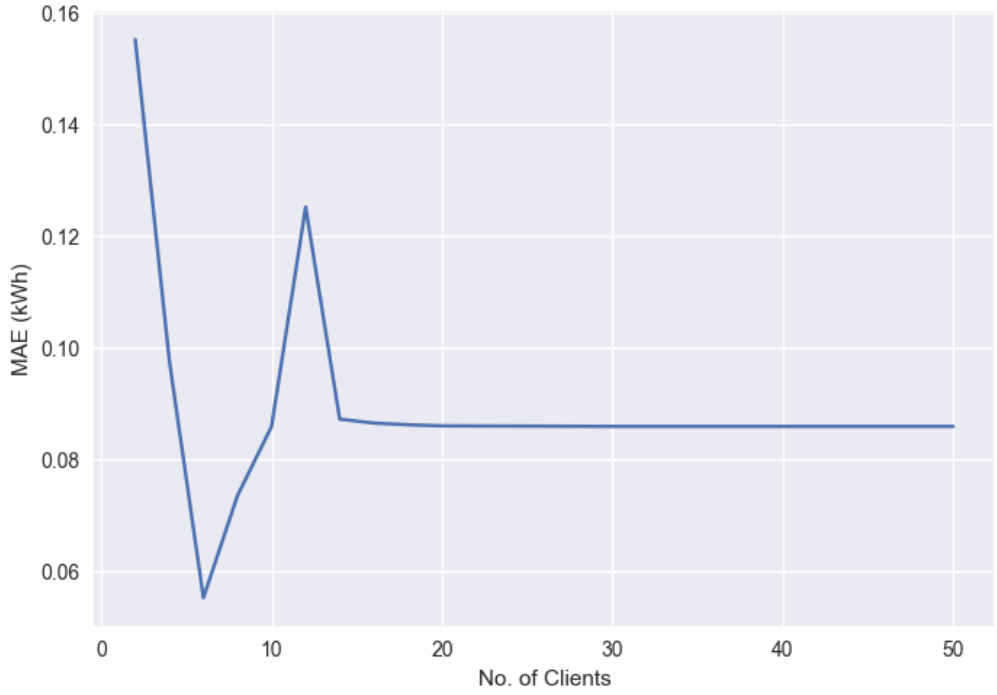


Figure 6.7. A comparison of the MAE for various customer numbers (participants).

6.3.4. Sensitivity to *N*-lags

Because this investigation is primarily concerned with the prediction of univariate energy consumption, our primary objective was to investigate the effect that the number of lags has on the overall performance and accuracy of the model. Surprisingly, the model's accuracy did not change at all even after we increased the lags from six to forty-eight hours. That is, the MAE and RMSE did not change no matter how far back we looked. In contrast, the time spent training or building is proportional to the number of lags. To optimize the model's overall performance, we opted to employ the smallest possible lags, which turned out to be six hours.

6.3.5. Centralized vs. decentralized (federated) architecture

The performance of the traditional centralized architecture and the federated architecture utilized in the proposed design in this study are compared in this subsection. In the first method, all the participants send their data to the server, which is run by the energy company. Next, a global model is trained using all the data sets that have been collected. Finally, the trained model is distributed to each participant so that they can make predictions about their local energy consumption. In contrast, every participant in federated learning keeps their

private data on-premises and receives a shared structure of the designed model from the energy company. This arrangement is known as distributed learning. After that, the process of training takes place simultaneously at each location. Then, the parameters of the constructed models, also known as weights, are sent to the server to be aggregated and transformed into a brand-new global model. In a manner analogous to that of the centralized architecture, the global model is disseminated to all participants (clients) to produce forecasts of later local energy consumption.

Table 6.4. Federated architecture vs. fog-cloud architecture.

Architecture	MAE	RMSE	Training time (min)
Adaptive federated	0.0859	0.1208	5.73
Adaptive centralized	0.1117	0.1310	30.46

Overall, as can be seen in Table 6.4, the performance of the federated architecture turned out to be superior to that of the centralized architecture. If we look at the RMSE, we see that the adaptive FL system reduced the amount of error in forecasting by almost 8%. In addition, it is important to note that the difference in training time between the two architectures shows that adaptive FL required less time to converge, 5.73 minutes as opposed to 30.46 minutes. The adaptive FL architecture is better than the current state-of-the-art architecture in more ways than we have already talked about. Some of these advantages are that user privacy is better protected and the amount of data sent over the network is lessened.

6.3.6. Adaptive federated learning: univariate vs multivariate models

In the field of mathematics, a univariate entity refers to an expression, equation, function, or polynomial that exclusively involves a single variable. Multivariate refers to objects that involve multiple variables. In the field of time series analysis, the term “variable” refers to the entire time series. Specifically, a univariate time series represents the sequence of values over a period of about a singular quantity. In a similar vein, a “multivariate time series” pertains to the dynamic evolution of multiple variables over a given period (Wan, Mei, Wang, Liu, & Yang, 2019).

In this subsection, we address the impact of using either a univariate or multivariate input architecture. In our case, the former is denoted as single input, single output (SISO), whereas the latter is multiple input, single output (MISO). In the preceding subsection, it was determined that stacked LSTM exhibits superior performance compared to BiLSTM in the context of SISO

adaptive federated learning. Consequently, following the engineering of the input features and incorporation of additional attributes such as calendar and weather data, experimentation was conducted to determine the optimal input-output architecture that would improve the forecasting accuracy of the proposed model.

The experimental findings indicate that the MISO-stacked LSTM model exhibits superior performance compared to the SISO model, resulting in a reduction of the forecasting error from 0.1208 to 0.0810 as shown in Table 6.5. Nevertheless, it is worth noting that the MISO technique requires a longer duration for training and convergence compared to the SISO approach.

Table 6.5. Univariate vs. multivariate input-output architecture for adaptive federated learning.

Model	input-output architecture	RMSE
stacked LSTM	SISO	0.1208
stacked LSTM	MISO	0.0810

6.3.7. Comparison with the Literature

We compared our suggested model to previous cutting-edge studies in the field of edge computing and federated learning, in general, and energy consumption, in particular, for comparison. As previously stated, few studies have addressed the application of federated learning to the topic of residential energy consumption, and to the best of our knowledge, none of these studies have considered adaptive learning as a complementary technique for this problem. Thus, we endeavor to combine adaptive learning with federated learning to improve the performance of the suggested model when applied to household energy consumption. So far, we have been able to compare our work with two previous models from the literature as shown in Table 6.6, which demonstrates the RMSE values achieved in those studies and ours.

In comparison, study (Taïk & Cherkaoui, 2020) forecast energy consumption using the univariate method and followed a similar methodology to our own. As shown in Table 6.6, their RMSE was roughly four times as large as ours. However, it is important to note that their data set is different from ours as they used 200 houses from Texas, USA while we used 100 houses from London, UK. On the other hand, study (Savi & Olivadese, 2021) utilized the same data set as we did, and our proposed model slightly outperformed theirs according to the RMSE metric.

In terms of how models were built, Savi et al. (Savi & Olivadese, 2021) used MISO architecture, whereas we used both SISO and MISO. Moreover, we attempted two variations of LSTM: stacked LSTM and Bi-LSTM to investigate which design achieves better outcomes with the problem at hand. Furthermore, our approach strengthens the proposed model with adaptive learning, which was not applied in either of the compared studies. Overall, Table 6.6 illustrates that our proposed model compared favorably to similar studies in the literature, and reduced the error rate by 9% when SISO is implemented compared to (Savi & Olivadese, 2021) using the same data set. Additionally, the rate of error reduction is enhanced to 39% when Multiple-Input Single-Output (MISO) is implemented in our proposed design of the adaptive federated learning system.

Table 6.6. A comparative analysis of performance with contemporary models.

Model	RMSE
Model proposed in (Taïk & Cherkaoui, 2020)	0.509
Model proposed in (Savi & Olivadese, 2021)	0.133
Our proposed model (SISO)	0.121
Our proposed model (MISO)	0.081

6.4. Conclusions

Federated learning architectures have been used in the literature to tackle privacy problems and network communication overhead. This work focuses on applying federated learning in an adaptive manner to the problem of energy consumption forecasting in smart cities. We designed and built robust stacked LSTM and BiLSTM recurrent neural network models. The former outperformed the latter in our experiments. We implemented a distributed architecture using a stacked LSTM model, incorporating adaptive learning techniques to enhance model performance and reduce daily learning requirements. Adaptivity improved architectural performance and reduced the root mean square error (RMSE) by 8\% in comparison to the standard federated learning (FL) approach. Moreover, when the input space was expanded by adding new features such as calendar and weather data, the experiments showed that the forecasting error could be reduced from 0.121 to 0.081 (RMSE), thereby improving the model's forecasting performance. The impact of participant count and lags on the accuracy of the proposed framework was found to be minimal. Therefore, we opted for smaller values to minimize training time complexity. During our investigation, it was observed that the model exhibited enhanced stability and robustness with an increase in the number of participants.

We compared our approach with the current centralized architecture. The adaptive federated architecture outperformed the adaptive centralized architecture, reducing training time by nearly 80% and improving accuracy by 8% and 38% for single-input-single-output (SISO) and multiple-input-single-output (MISO), respectively. However, certain areas and aspects require additional clarification. In our future work, we plan to thoroughly investigate significant challenges such as privacy protection, network communication overhead, and non-IID (non-independent and identically distributed) data in the context of federated learning. Also, to ensure a comprehensive analysis of the impact of adaptive learning across various seasons and data drifts, we aim to extend the duration of both the training and testing set to a two-year timeframe. Subsequently, the suggested architecture can be employed to analyze another recent data set to assess its robustness.

7. DISCUSSION

The aim of this research is to examine the effectiveness of incorporating adaptive learning into distributed machine learning architectures for the purpose of data analytics and time series forecasting in diverse smart city applications. In the preceding chapters (from study 1 to study 3), we systematically examined the suggested system across various applications and architectures, elucidating its advantages and disadvantages. The subsequent paragraphs highlight the principal discoveries and significant knowledge acquired during this investigation. The shared characteristic across smart city applications such as temperature, flood, and household energy consumption forecasting is that they all rely on time-series data. Temperature, flood levels, and household energy usage display discernible patterns and trends as time progresses, making time-series forecasting a suitable methodology.

Conversely, the following are possible distinctions among these three applications:

1. Characteristics of the variable:
- "Tempest forecasting" is the process of estimating the future atmospheric temperatures at different locations.
- In flood forecasting, water levels in rivers and other bodies of water are predicted, frequently because of precipitation or other meteorological phenomena.
- The objective of household energy consumption forecasting is to estimate the temporal variation in energy usage among households, considering various factors such as weather conditions, occupancy trends, and appliance utilization.
2. The sources of data:
- Temperature forecasting is dependent on meteorological data encompassing historical temperature measurements, wind speed, atmospheric pressure, and humidity.
- Utilizes information including topography, precipitation patterns, river levels, and soil moisture content to forecast flooding.
- Forecasting household energy consumption requires information regarding past energy usage trends, meteorological conditions, architectural attributes, and occupant conduct.
3. Objectives of forecasting:
- Temperature forecasting plays a crucial role in providing weather information to industries such as agriculture, public safety, and others.

- The objective of flood forecasting is to manage the effects of flooding on human settlements, agriculture, and infrastructure by providing early warnings.
- Forecasting household energy consumption assists utilities and consumers in optimizing energy usage, preparing for fluctuations in demand, and promoting energy efficiency.

4. Lead time and response time:

- Temperature predictions for the short to medium term are utilized in the organization of daily activities and events.
- Flood forecasting is a critical process that generally entails making short-term predictions to issue early warnings, facilitate opportune evacuation, or safeguard infrastructure.
- Forecasting household energy consumption for load balancing, grid management, and energy demand planning over the short to medium term.

5. External factors of influence:

- Temperature forecasting is subject to the impact of meteorological phenomena and atmospheric conditions.
- Flood forecasting is influenced by various hydrological factors, including precipitation and snowmelt.
- The prediction of household energy consumption is not only impacted by meteorological conditions, but also by human conduct, appliance utilization, and architectural attributes.

6. Stakeholders participating:

- Stakeholders in temperature forecasting comprise emergency services, agriculture, transportation, and meteorological agencies.
- Flood forecasting is a complex undertaking that demands the cooperation of local governments, emergency services, meteorological agencies, and water resource authorities.
- Stakeholders in the domain of household energy consumption forecasting comprise energy regulators, utility corporations, and consumers.

Gaining an understanding of these prospective distinctions enables the creation of customized time-series forecasting models that effectively tackle the distinct obstacles and demands of individual smart city applications.

The authors propose using adaptive learning approaches in distributed machine learning (ML) and deep learning (DL) models to improve performance and prediction accuracy in dealing with constantly changing data streams from IoT devices. ML enables systems to acquire information from historical or current data, while DL offers automated feature extraction, eliminating the need for manual effort. DL has gained popularity as the IoT-based domain due to its large volume of data. In addition, this study investigates two distributed learning architectures, namely fog-cloud computing and federated learning. The advantages and disadvantages of each architecture, as well as their most suitable application scenarios, are examined. Additionally, we utilize weakly supervised learning techniques to perform data annotation and augmentation as a preprocessing step inside our system design. This approach allows us to effectively leverage the vast amount of unlabeled data or noisy data generated by IoT devices for training our models. Therefore, the integration of these various components yields a complex system design that ensures enhanced precision in forecasting, reduced rates of error, and decreased time required for retraining and data transmission between edge devices and central servers.

In the beginning, it was shown through conducted experiments that deep learning models exhibit superior performance compared to machine learning models, particularly in scenarios where there is a substantial volume of training data available. To provide further clarification, when the number of data points (samples) in a data set surpasses ten thousand, it is strongly advised to utilize deep learning models. Otherwise, it is more advantageous to adhere to machine learning models. The rationale for this phenomenon lies in the inherent complexity and depth of deep learning algorithms, which necessitate an enormous quantity of training data in order to achieve optimal learning outcomes. Alternatively, the responsibility might be delegated to shallow learning, specifically machine learning, in order to mitigate the issue of overfitting. Additionally, in the case of machine learning, a reduced amount of time and computational resources are utilized. These findings are consistent with existing literature and widely accepted information, as exemplified in (Cho et al., 2020; Cioffi et al., 2020; Fekri et al., 2021; Hackeling, 2017; Y. Liu et al., 2023), to name a few.

In addition to comparing ML and DL models' performance, we explore several variations of deep learning algorithms as well to select the most optimal model for our time series forecasting applications. One of the applications (studies) of this thesis is to investigate the problem of time series data in weather forecasting by employing Recurrent Neural Network

(RNN) deep learning models, with a specific focus on the Long Short-Term Memory (LSTM) model. The study examined three different architectural models, namely stacked LSTM, bidirectional LSTM, and CNN LSTM. The findings of the study indicate that the implementation of bidirectional LSTM resulted in a notable decrease of approximately 31% in the error prediction rate, when compared to the most effective Machine Learning (ML) model. The researchers conducted experiments to assess the resilience of LSTM models in both univariate and multivariate prediction scenarios, employing various time steps (lookbacks). The findings indicated that the implementation of the univariate model resulted in improved prediction accuracy for all LSTM models. As the number of lookbacks increases, there is a corresponding increase in the root mean square error (RMSE) value. Conversely, in the context of multivariate mode, a decrease in the number of lookbacks was shown to yield superior results.

Having that said, the unencouraging output from the multivariate model could be attributed to the low correlation among the other weather features such as humidity, rainfall and the output variable which is temperature. Table 4.4 shows the various accuracies acquired by univariate and multivariate models. It is recommended that clean feature engineering and selection is applied to the data set to lead to promising results and more accurate forecasting. This recommendation is carried out in the third study (household energy consumption) and the outcomes prove that using well-designed feature engineering techniques could boost the model skills by adding the appropriate features. This can be shown in Table 6.5. However, if the univariate approach maintains sufficient accuracy, it is preferable due to its lower computational requirements and resources usage. Consequently, it leads to a lightweight and responsive model, which is very desirable for applications that require near real-time performance.

One of the key questions in this research is that does implement adaptive learning to the time series forecasting enhance the model skill and accuracy? Thus, this thesis additionally examined the efficacy of adaptive learning on deep learning models by employing a sample size of 11,000 from the IMM data set for temperature forecasting, 12,00 data points for flood forecasting, and more than 200 thousand samples for energy consumption forecasting application. The findings indicated that the adaptive model exhibited superior accuracy compared to the baseline model, resulting in a reduction of the total error prediction rate by around 20% as shown in Table 4.7 for temperature forecasting, nearly 12% as shown in

Figure 5.6 for flood forecasting, and about 7% as shown in Table 6.3 for energy consumption forecasting. According to the literature, the listed smart city applications exhibit substantial improvement rates. It is advisable to adopt these applications if the model indicates an improvement of more than 5% (Chen et al., 2021; Cifuentes et al., 2020; Savi & Olivadese, 2021). Since the least improvement we have is 7% in the flood forecasting case study, exploiting these structures in future applications is recommended. The investigations and smart city applications done in this thesis demonstrate that the incorporation of the adaptive learning concept enhances the performance of the model. This improvement is characterized by an increase in accuracy and a reduction in the frequency of retraining. According to our proposed system, the model undergoes retraining exclusively with recent historical data in the event of detecting a concept drift or data drift. Consequently, the system exhibits a higher level of sophistication and robustness compared to the conventional retraining approach. The utilization of the suggested model results in the conservation of system resources, including power and memory processing. This is achieved by exclusively training the model solely when its performance deteriorates because of the presence of data or concept drift. Furthermore, in the scenario of using distributed computation, less data transmission happens for the model update.

Furthermore, this thesis aims to study the pros and cons of applying distributed computation compared to centralized computation which represents in either cloud computing or fog computing. Accordingly, we contrast central computing with two decentralized architectures called fog-cloud computing and federated learning (FL). We embarked on fog-cloud computing in the application of temperature and weather forecasting for smart cities. In terms of model performance, both centralized and decentralized architectures show identical accuracy. However,

Table 4.2 presents a comprehensive overview of the training durations, measured in seconds, for a comparable adaptive model deployed on both fog and cloud nodes. The training set is intentionally augmented to observe the impact on training times. Based on the data presented in

Table 4.2, it is evident that the training time for the model is significantly reduced when utilizing cloud infrastructure, as previously discussed, due to the cloud's superior computational capabilities. Moreover, a notable observation from the table is the occurrence

of fog node collapse when the data set exceeds 9,000 units. This finding provides evidence that the fog node is incapable of effectively managing operations on a large scale. So, it becomes evident that the proposed model of adaptive learning on a collaborative fog-cloud architecture for streaming data yields noteworthy results in terms of forecasting accuracy, training speed of the model, and scalability compared to only using fog computing (centralized architecture). Moreover, the performance of the models significantly improves when they are trained on cloud platforms and subsequently deployed on fog nodes for the purpose of forecasting responsiveness in near real-time applications.

On the other hand, when federated learning is compared to centralized learning, different insights were concluded. For instance, it is evident from the data presented in Table 6.4 that the federated design exhibited greater performance compared to the centralized architecture. Upon examination of the root mean square error (RMSE), it is distinct that the adaptive FL system achieved a nearly 8% reduction in forecasting error. Furthermore, it is noteworthy to acknowledge that the disparity in training duration between the two architectures indicates that adaptive FL necessitated a less amount of time to reach convergence, specifically 5.73 minutes compared to 30.46 minutes. Also, the adaptive FL design exhibits superior performance compared to the existing state-of-the-art architecture, surpassing the aspects we have previously discussed, including enhanced protection of user privacy and less transmission of data across the network. In summary, both distributed architectures, namely fog-cloud computing, and federated learning (FL), are strongly recommended for time series forecasting in smart city applications. This is due to their ability to enhance training data generalization, resulting in improved accuracy. Additionally, these architectures effectively expedite the training process by efficiently utilizing appropriate resources. Furthermore, they support the scalability of the vast amount of big data generated from IoT-based applications.

During the course of our thesis, while doing various smart city time series forecasting analyses, we encountered relevant literature pertaining to the three applications we have chosen to implement. In order to ascertain the validity of their insights or demonstrate the advancements achieved through the utilization of our recommended system design for a certain application, it is customary to engage in a comparative analysis of their findings with our own. At the outset, when contrasted with contemporary and cutting-edge research endeavors that focus on temperature prediction and the use of LSTM-based models, the optimal adaptive bidirectional model developed in this work demonstrates a superiority of

over 50% compared to recently proposed models in the same domain. It is important to note that the data sets utilized in these three studies differ, although having a similar duration of less than two years for training and testing. However, attaining the same benchmark and evaluating the various proposed models in a consistent manner will yield more definitive insights and promote more equitable comparisons.

Additionally, our objective is to verify the validity of the proposed model for temperature forecasting by implementing it on various datasets to assess whether the adaptive model will continue to yield improved prediction accuracy. Therefore, a concise overview of these investigations is provided in Figure 4.6. As anticipated, the adaptive bidirectional LSTM model consistently exhibits a lower number of prediction errors in relation to the root mean square error (RMSE). However, it is worth noting that the model achieves the lowest Root Mean Square Error (RMSE) values at different time steps for the three datasets. The rationale behind this phenomenon can be ascribed to the following: when the data points exhibit non-stationarity and high volatility, it is advisable to reduce the number of time-steps considered to obtain a more accurate analysis. To clarify, the Beijing PM2.5 data set exhibits greater variability in temperature values compared to the other data sets that have been utilized. This implies that various datasets within the same domain invariably require human intervention to adjust the model settings and hyperparameters to get optimal outcomes.

Regarding the FL architecture applied to the household energy consumption forecasting, we conducted a comparative analysis between our proposed system architecture and existing state-of-the-art research in the domains of fog-cloud computing and federated learning, with a specific focus on energy usage. As previously mentioned, there is a limited body of research that has explored the implementation of federated learning in the context of residential energy usage. Furthermore, to the best of our knowledge, none of these studies have examined the potential of adaptive learning as a supplementary approach to solve this issue. Therefore, our objective is to integrate adaptive learning and federated learning methodologies to enhance the efficacy of the proposed model in the context of residential energy usage. To this end, we have conducted a comparative analysis of our research with two existing models documented in the literature. The results of this analysis, presented in Table 6.6, illustrate the respective root mean square error (RMSE) values attained in the aforementioned studies as well as our own. In contrast, a study conducted by Taik et al. (2020) (Taïk & Cherkaoui, 2020) employed the univariate method to predict energy usage

and adopted a methodology that closely resembles our own. The root mean square error (RMSE) of their results was almost four times greater than that of our findings. It is crucial to acknowledge that there exists a disparity between their dataset and ours, as their study encompassed a sample size of 200 residences located in Texas, USA, whilst our investigation focused on a sample size of 100 houses situated in London, UK. On the other hand, a previous work by Savi et al. (2021) (Savi & Olivadese, 2021) employed the identical dataset as our study. Our proposed model shown a somewhat superior performance compared to theirs, as evidenced by the RMSE measure. Regarding the construction of models, the study conducted by (Savi & Olivadese, 2021) employed a MISO architecture, whereas our approach incorporated both SISO and MISO architectures. In addition, we conducted experiments with two versions of LSTM, namely stacked LSTM and Bi-LSTM, to assess their respective efficacy in addressing the given problem. Additionally, our methodology enhances the suggested model through the implementation of adaptive learning, a feature that was not utilized in either of the research being examined. In general, the findings presented in Table 6.6 demonstrate that our proposed model exhibits favorable performance in comparison to previous research in the field. Specifically, when using a single-input single-output (SISO) approach, our model achieves a notable 9% reduction in error rate when compared to a study conducted by Savi et al. (Savi & Olivadese, 2021) that utilized the same dataset. Furthermore, the implementation of Multiple-Input Single-Output (MISO) in our suggested framework leads to a more enhanced rate of error reduction that reached 39%.

Overall, this thesis's designed architecture and developed adaptive models illustrate the superiority of using adaptive learning as an integral component for time-series forecasting. Furthermore, we proved that applying distributed machine learning methods in decentralized architectures like fog-cloud computing and federated learning increases the model's skills, scalability, and data privacy while decreasing the model's construction time, data exchange over the network, and responsiveness. The improvement accomplished for the three studies (applications) varies from application to application. That is, we achieve higher performance for temperature and energy consumption problems than flood forecasting. Nevertheless, this could be attributed to the low quality and low dimensionality of the input space for the latter while the rich problem space for the former. In other words, we only rely on rainfall features as input for forecasting flood occurrences, which is a very convoluted problem and needs more sophisticated input features to reach satisfactory accuracy for the stakeholders. In the

introduction section, we have already mentioned our justification and explanation, which will be considered a baseline for future studies.

Like any other scholarly inquiry, this study has inherent limitations. Investigating time-series forecasting in multiple smart city applications utilizing distributed and adaptive machine learning models is the primary objective of this work. It is critical to mention, nevertheless, that the data sets utilized in this investigation might be inadequate in scope, as they solely encompass a temporal duration of no more than two years. This span may not be sufficient for deep learning techniques that require voluminous amounts of data, such as LSTM models. The proposition that a model's performance enhances with prolonged exposure to a larger volume of data points is supported by empirical evidence. Hence, it might be practical to contemplate employing a more extensive data set in future research endeavors in order to augment the representativeness of the results. Moreover, the scarcity of comparative studies employing the exact same data set or focusing solely on a specific application forecasting issue restricts the potential for comparing findings with state-of-the-art research. However, the authors tried to integrate the prevalent models that are frequently employed in the current corpus of research into this investigation. This was performed to compare these models to the proposed model, thereby showcasing its efficacy.

Moreover, the researchers encountered a limitation in the form of restricted access to data sets from previous studies, which were not available to the public. In summary, it is expected that forthcoming research designs will place a more significant emphasis on comparative analysis. The authors aim to enhance and deploy the proposed model on widely recognized and easily accessible benchmarks so that its performance can be assessed in relation to cutting-edge algorithms and methodologies. Furthermore, one could argue that the extent of improvement is negligible in certain contexts where the adaptive model was implemented, considering the additional memory usage, processing power, and communication bandwidth necessary for retraining the model after data drift is detected. There are two possible explanations for the reduced improvement observed in this experiment because of adaptive learning. Initially, data drift was prevented by the restricted evaluation period in December 2013, given that residential energy consumption tends to be stable during the winter. Consequently, it is unnecessary to revise the model using the algorithm that we have proposed. The resultant consequence was a marginal improvement in the performance of the model. Furthermore, it is expected that the effectiveness of adaptability will be demonstrated

as the testing phase is extended to encompass a wider array of months and seasons. Hence, when neither concept drift nor data drift occurs, the practicality of adaptive learning in the domain of time-series data may be limited. However, it is critical to acknowledge that these types of anomalies are commonly detected in applications involving time series. Hence, we contend that the application of adaptive learning techniques is advantageous in this setting. Besides, thus far, the investigation has been conducted utilizing genuine data, though in a simulated fashion, by implementing infrastructure and software such as the TensorFlow federated learning API and the Apache Spark platform. The outcomes demonstrate potential and warrant verification through the implementation of these concepts in tangible, real-life scenarios. Enhanced value will be derived from this implementation of the proposed paradigm and architecture.

8. CONCLUSION AND RECOMMENDATIONS

In brief, this scientific doctoral dissertation investigates a multitude of adaptive learning applications across distinct domains, encompassing temperature prediction, flood forecasting, energy consumption forecasting, and streaming data processing. The study provides evidence that the incorporation of adaptive learning into machine learning algorithms substantially improves the accuracy of predictions and diminishes error rates. The integration of fog-cloud architecture and federated learning architecture with adaptive learning results in additional enhancements to overall performance and reductions in forecasting process latency.

8.1. Conclusion

So, this thesis examines the utilization of adaptive learning techniques in the context of streaming data processing, with the aim of enhancing prediction models in comparison to traditional machine learning algorithms. Additionally, the study emphasizes the integration of distributed methods and learning such as fog-cloud architecture and federated learning, which effectively improves overall system performance and mitigates latency during the prediction process. The findings of the first application suggest that the random forest approach demonstrates superior performance in regression tasks pertaining to temperature prediction. Additionally, it is shown that periodically updating the model yields improved accuracy in forecasting temperature. The implementation of the adaptive model also results in a notable decrease in the rate of prediction errors, amounting to a reduction of around 20%. Furthermore, during the training process of the adaptive model on the cloud layer, there is a reduced amount of time required for training and improved scalability. The results of this study indicate that training the model on the cloud layer and subsequently deploying it to fog nodes for prediction can enhance the accuracy of predictions and reduce latency in terms of prediction speed and delivery to IoT devices.

Subsequently, the study additionally examines the utilization of adaptive learning in conjunction with a bidirectional Long Short-Term Memory (LSTM) model for the purpose of time-series forecasting. The implementation of cloud computing in the construction of the ultimate adaptive model results in a significant reduction of around 45% in the prediction error rate. Additionally, the time necessary for creating the model can be decreased by

approximately 85%. When the suggested model is implemented on several time-series meteorological datasets, the Mean Absolute Percentage Error (MAPE) consistently falls within the range of 8-10%. This suggests that the model exhibits both robustness and effectiveness. The research also places emphasis on the prediction of flood occurrences by exclusively utilizing meteorological data, with a specific focus on rainfall patterns. Weakly supervised learning approaches are employed for the preprocessing of datasets characterized by their limited size and the presence of noisy or incomplete labels. The functions of labeling and transformation serve the purposes of assigning labels to missing classes and expanding the size of the data set, respectively, resulting in larger and annotated samples without the intervention of human being. This reduces the cost of expertise for data labeling and speeds up the labeling process.

Finally, federated learning architecture are employed to address concerns regarding privacy and the excessive overhead associated with network communication. The primary objective of this study is to investigate the adaptive implementation of federated learning for the purpose of energy consumption forecasting in smart cities. The researchers developed and constructed resilient stacked LSTM and BiLSTM recurrent neural network models, where the former demonstrated superior performance compared to the latter in experimental trials. The results indicate that the adaptive federated architecture illustrated better performance compared to the adaptive centralized design. Specifically, it significantly reduced the training time by almost 80% and achieved noteworthy improvements in accuracy, with enhancements of 8% and 38% for single-input-single-output (SISO) and multiple-input-single-output (MISO) scenarios, respectively.

A summary of how the three study cases connect to the smart city context follows. Precise temperature forecasting, for example, is essential for energy efficiency optimization, urban infrastructure management, and preparation for weather-related occurrences. By incorporating temperature predictions into the municipality's energy management systems, the effectiveness of heating, ventilation, and air conditioning (HVAC) systems can be improved, fostering energy conservation and cost efficiency. In contrast, flood forecasting is a critical component of Smart City resilience as it aids in protecting infrastructure and citizens while mitigating the effects of extreme weather. IoT sensor deployment in flood-prone regions permits data collection in real-time. The integration of flood forecasting models enables the municipality to promptly issue alerts, optimize evacuation routes, and

execute preventive measures to mitigate flood-induced damages. Lastly, energy consumption forecasting for households. Energy efficiency is a fundamental objective of smart cities—predictions of residential energy consumption support demand-side management and sustainable resource utilization. The city can optimize overall energy distribution, implement dynamic pricing strategies, and incentivize energy-efficient practices by implementing predictive analytics for domestic energy consumption.

In each of the three forecasting scenarios, integrating data from various sources is crucial, underscoring the importance of a robust data infrastructure within a Smart City. A unified data platform strengthens a municipality's resilience and overall operations by facilitating holistic decision-making. In addition, IoT-enabled real-time monitoring is critical in every case study for timely intervention and decision-making. It facilitates the city's capacity to proactively address changing circumstances, enhancing resource allocation and public safety. In addition, forecasting models optimize resource allocation and planning by utilizing predictive analytics to anticipate unanticipated future events. The utilization of predictive analytics in Smart City systems enables decision-makers to take a proactive approach, reducing risks and maximizing the effective allocation of resources. All in all, the utilization and execution of energy, flood, and temperature prediction systems contribute to the overarching objective of developing urban environments that are more sustainable, resilient, and efficient.

To conclude, we have answered the research questions in the thesis and summarize them as follows:

- Integrating adaptive learning into a fog-cloud architecture that operates collaboratively potentially increases the accuracy of prediction models. It reduces the latency of real-time data processing in Internet of Things applications.
- When large data scenarios arise, the performance of time-series forecasting models is enhanced when a variant of deep learning models is utilized instead of cutting-edge machine learning models.
- The model's performance could be enhanced by combining adaptive learning with weakly supervised learning for data augmentation and labeling to account for seasonality in time series forecasting.

- When adaptive learning and federated learning are combined, predictive model accuracy is enhanced, privacy is preserved more effectively, and training time is reduced compared to fog-cloud computing architecture.

8.2. Recommendations

To further enhance the scope of this study, it is recommended to incorporate a published flood forecast dataset and conduct a comparative analysis between the proposed methodology and its corresponding outcomes. Additionally, it is possible to incorporate and analyze new flood disaster data and features. Another potential avenue to consider is the transition from centralized computation to distributed learning. Ultimately, the proposed model has the potential to be implemented in a different smart city context, specifically one that involves the utilization of time series streaming datasets.

Additionally, there are specific regions and elements that want further elucidation. In forthcoming research endeavors, we encourage conducting a comprehensive examination of prominent obstacles, including privacy preservation, network communication costs, and non-IID (non-independent and identically distributed) data, within the domain of federated learning. To conduct a thorough examination of the effects of adaptive learning in different seasons and with varying data drifts, the objective is to expand the duration of both the training and testing sets to encompass a period of two years. Following this, the proposed system architecture can be utilized to examine a different, more recent dataset with the purpose of evaluating its resilience. Moreover, the weakly supervised learning techniques for automatic data labeling and augmentation could be employed as a preprocessing step for the proposed system architecture so that missing labels and noisy data issues are solved out.

REFERENCES

Aazam, M., and Huh, E.-N. (2015). Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT. *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, 687–694. IEEE.

Aburukba, R. O., AliKarrar, M., Landolsi, T., and El-Fakih, K. (2020). Scheduling Internet of Things requests to minimize latency in hybrid Fog–Cloud computing. *Future Generation Computer Systems*, *111*, 539–551.

Alam, T. (2021). Cloud-Based IoT Applications and Their Roles in Smart Cities. *Smart Cities*, *4*(3), 1196–1219. https://doi.org/10.3390/smartcities4030064

Alli, A. A., and Alam, M. M. (2020). The fog cloud of things: A survey on concepts, architecture, standards, tools, and applications. *Internet of Things*, *9*, 100177.

Almagbile, A., and Hazaymeh, K. (2022). Spatiotemporal variability/stability analysis of NO2, CO, and land surface temperature (LST) during COVID-19 lockdown in Amman city, Jordan. *Geo-Spatial Information Science*, 1–18.

Alonso, A. M., Nogales, F. J., and Ruiz, C. (2020). A single scalable LSTM model for short-term forecasting of massive electricity time series. *Energies*, *13*(20), 5328.

Al-Qaness, M. A., Fan, H., Ewees, A. A., Yousri, D., and Abd Elaziz, M. (2021). Improved ANFIS model for forecasting Wuhan City Air Quality and analysis COVID-19 lockdown impacts on air quality. *Environmental Research*, *194*, 110607.

Amasyali, K., and El-Gohary, N. M. (2018). A review of data-driven building energy consumption prediction studies. *Renewable and Sustainable Energy Reviews*, *81*, 1192–1205.

Apache Spark™—Unified Engine for large-scale data analytics. (n.d.). Retrieved October 26, 2023, from https://spark.apache.org/

Asghar, M. R., Dán, G., Miorandi, D., and Chlamtac, I. (2017). Smart meter data privacy: A survey. *IEEE Communications Surveys & Tutorials*, *19*(4), 2820–2835.

Aurangzeb, K., Aslam, S., Haider, S. I., Mohsin, S. M., Islam, S. ul, Khattak, H. A., & Shah, S. (2022). Energy forecasting using multiheaded convolutional neural networks in efficient renewable energy resources equipped with energy storage system. *Transactions on Emerging Telecommunications Technologies*, *33*(2), e3837.

Birch, C. E., Rabb, B. L., Böing, S. J., Shelton, K. L., Lamb, R., Hunter, N., and Dale, M. (2021). Enhanced surface water flood forecasts: User-led development and testing. *Journal of Flood Risk Management*, *14*(2), e12691. https://doi.org/10.1111/jfr3.12691

Bittencourt, L. F., Lopes, M. M., Petri, I., and Rana, O. F. (2015). Towards virtual machine migration in fog computing. *2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, 1–8. IEEE.

Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., and McMahan, B. (2019). Towards federated learning at scale: System design. *Proceedings of Machine Learning and Systems*, *1*, 374–388.

Brownlee, J. (2018). A gentle introduction to concept drift in machine learning. *Machine Learning Mastery*.

Chegini, H., Naha, R. K., Mahanti, A., and Thulasiraman, P. (2021). Process automation in an IoT–fog–cloud ecosystem: A survey and taxonomy. *IoT*, *2*(1), 92–118.

Chen, C., Hui, Q., Xie, W., Wan, S., Zhou, Y., and Pei, Q. (2021). Convolutional Neural Networks for forecasting flood process in Internet-of-Things enabled smart city. *Computer Networks*, *186*, 107744. https://doi.org/10.1016/j.comnet.2020.107744

Chitsaz, H., Shaker, H., Zareipour, H., Wood, D., and Amjady, N. (2015). Short-term electricity load forecasting of buildings in microgrids. *Energy and Buildings*, *99*, 50–60.

Cho, D., Yoo, C., Im, J., and Cha, D.-H. (2020). Comparative assessment of various machine learning-based bias correction methods for numerical weather prediction model forecasts of extreme air temperatures in urban areas. *Earth and Space Science*, *7*(4), e2019EA000740.

Cifuentes, J., Marulanda, G., Bello, A., and Reneses, J. (2020). Air Temperature Forecasting Using Machine Learning Techniques: A Review. *Energies*, *13*(16), 4215. https://doi.org/10.3390/en13164215

Cioffi, R., Travaglioni, M., Piscitelli, G., Petrillo, A., and De Felice, F. (2020). Artificial intelligence and machine learning applications in smart production: Progress, trends, and directions. *Sustainability*, *12*(2), 492.

Dai, W., Tang, Y., Zhang, Z., and Cai, Z. (2021). Ensemble learning technology for coastal flood forecasting in internet-of-things-enabled smart city. *International Journal of Computational Intelligence Systems*, *14*, 1–16.

Das, M., and Ghosh, S. K. (2018). Data-driven approaches for meteorological time series prediction: A comparative study of the state-of-the-art computational intelligence techniques. *Pattern Recognition Letters*, *105*, 155–164.

Das, T., Zhong, Y., Stoica, I., and Shenker, S. (2014). Adaptive stream processing using dynamic batch sizing. *Proceedings of the ACM Symposium on Cloud Computing*, 1–13.

DataStore, L. (2018). *Smartmeter energy consumption data in london households*.

Deshmukh, U. A., and More, S. A. (2016). Fog computing: A new approach in the world of cloud computing. *Instr Technol*, *49*.

Diao, E., Ding, J., and Tarokh, V. (2021). *SemiFL: Communication efficient semi-supervised federated learning with unlabeled clients*.

Fallah, S. N., Deo, R. C., Shojafar, M., Conti, M., and Shamshirband, S. (2018). Computational Intelligence Approaches for Energy Load Forecasting in Smart Energy Management Grids: State of the Art, Future Challenges, and Research Directions. *Energies*, *11*(3), 596. https://doi.org/10.3390/en11030596

Fan, Z., Kalogridis, G., Efthymiou, C., Sooriyabandara, M., Serizawa, M., and McGeehan, J. (2010). The new frontier of communications research: Smart grid and smart metering. *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, 115–118.

Fekri, M. N., Patel, H., Grolinger, K., and Sharma, V. (2021). Deep learning for load forecasting with smart meter data: Online Adaptive Recurrent Neural Network. *Applied Energy*, *282*, 116177.

Hackeling, G. (2017). *Mastering Machine Learning with scikit-learn*. Packt Publishing Ltd.

Hayati, M., and Mohebi, Z. (2007). *Application of Artificial Neural Networks for Temperature Forecasting*.

Hewage, P., Behera, A., Trovati, M., Pereira, E., Ghahremani, M., Palmieri, F., and Liu, Y. (2020). Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station. *Soft Computing*, *24*(21), 16453–16482. https://doi.org/10.1007/s00500-020-04954-0

Hewage, P., Trovati, M., Pereira, E., and Behera, A. (2021). Deep learning-based effective fine-grained weather forecasting model. *Pattern Analysis and Applications*, *24*(1), 343–366.

Hingmire, A. M., and Bhaladhare, P. R. (2022). A review on urban flood management techniques for the smart city and future research. *International Conference on Intelligent Cyber Physical Systems and Internet of Things*, 303–317. Springer.

Hong, Y., Zhou, Y., Li, Q., Xu, W., and Zheng, X. (2020). A deep learning method for short-term residential load forecasting in smart grid. *IEEE Access*, *8*, 55785–55797.

Hu, Y. C., Patel, M., Sabella, D., Sprecher, N., and Young, V. (2015). Mobile edge computing—A key technology towards 5G. *ETSI White Paper*, *11*(11), 1–16.

Hyndman, R. J. (2006). Another look at forecast-accuracy metrics for intermittent demand. *Foresight: The International Journal of Applied Forecasting*, *4*(4), 43–46.

Kang, Y., Liu, Y., and Liang, X. (2022). FedCVT: Semi-supervised vertical federated learning with cross-view training. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *13*(4), 1–16.

Kassem, H., Alapatt, D., Mascagni, P., AI4SafeChole, C., Karargyris, A., & Padoy, N. (2022). Federated cycling (FedCy): Semi-supervised Federated Learning of surgical phases. *IEEE Transactions on Medical Imaging*.

Kaytez, F., Taplamacioglu, M. C., Cam, E., and Hardalac, F. (2015). Forecasting electricity consumption: A comparison of regression analysis, neural networks and least squares

support vector machines. *International Journal of Electrical Power & Energy Systems*, *67*, 431–438.

Khan, A. R., Mahmood, A., Safdar, A., Khan, Z. A., and Khan, N. A. (2016). Load forecasting, dynamic pricing and DSM in smart grid: A review. *Renewable and Sustainable Energy Reviews*, *54*, 1311–1322. https://doi.org/10.1016/j.rser.2015.10.117

Khan, W. Z., Ahmed, E., Hakak, S., Yaqoob, I., and Ahmed, A. (2019). Edge computing: A survey. *Future Generation Computer Systems*, *97*, 219–235.

Kim, J. Z., Lu, Z., Nozari, E., Pappas, G. J., and Bassett, D. S. (2021). Teaching recurrent neural networks to infer global temporal structure from local examples. *Nature Machine Intelligence*, *3*(4), 316–323. https://doi.org/10.1038/s42256-021-00321-2

Kim, T. K. (2015). T test as a parametric statistic. *Korean Journal of Anesthesiology*, *68*(6), 540–546. https://doi.org/10.4097/kjae.2015.68.6.540

Kim, T., Ramos, C., and Mohammed, S. (2017). Smart city and IoT. In *Future Generation Computer Systems* (Vol. 76, pp. 159–162). Elsevier.

Ku, J.-H. (2018). A study on adaptive learning model for performance improvement of stream analytics. *Journal of Convergence for Information Technology*, *8*(1), 201–206.

Kuncheva, L. I. (2008). Classifier ensembles for detecting concept change in streaming data: Overview and perspectives. *2nd Workshop SUEMA*, *2008*, 5–10.

Li, H., Wang, Y., Wang, H., and Zhou, B. (2017). Multi-window based ensemble learning for classification of imbalanced streaming data. *World Wide Web*, *20*, 1507–1525.

Li, Y., Zhu, Z., Kong, D., Han, H., and Zhao, Y. (2019). EA-LSTM: Evolutionary attention-based LSTM for time series prediction. *Knowledge-Based Systems*, *181*, 104785.

Liang, X., Zou, T., Guo, B., Li, S., Zhang, H., Zhang, S., and Chen, S. X. (2015). Assessing Beijing's PM2. 5 pollution: Severity, weather impact, APEC and winter heating. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *471*(2182), 20150257.

Liu, F., Zeng, C., Zhang, L., Zhou, Y., Mu, Q., Zhang, Y., and Zhu, C. (2022). FedTADBench: Federated Time-series Anomaly Detection Benchmark. *2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*, 303–310. IEEE.

Liu, J., Zhang, T., Han, G., and Gou, Y. (2018). TD-LSTM: Temporal dependence-based LSTM networks for marine temperature prediction. *Sensors*, *18*(11), 3797.

Liu, X., Zhu, L., Xia, S.-T., Jiang, Y., and Yang, X. (2021). GDST: Global Distillation Self-Training for Semi-Supervised Federated Learning. *2021 IEEE Global Communications Conference (GLOBECOM)*, 1–6. IEEE.

Liu, Y., Zhou, Y., Yang, K., and Wang, X. (2023). Unsupervised Deep Learning for IoT Time Series. *IEEE Internet of Things Journal*.

Lynch, P. (2006). *The Emergence of Numerical Weather Prediction: Richardson's Dream*. Cambridge University Press.

Maiti, P., Shukla, J., Sahoo, B., and Turuk, A. K. (2018). QoS-aware fog nodes placement. *2018 4th International Conference on Recent Advances in Information Technology (RAIT)*, 1–6. IEEE.

Mármol, F. G., Sorge, C., Ugus, O., and Pérez, G. M. (2012). Do not snoop my habits: Preserving privacy in the smart grid. *IEEE Communications Magazine*, *50*(5), 166–172.

Martin, B. A., Michaud, F., Banks, D., Mosenia, A., Zolfonoon, R., Irwan, S., and Zao, J. K. (2017). OpenFog security requirements and approaches. *2017 IEEE Fog World Congress (FWC)*, 1–6. IEEE.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. *Artificial Intelligence and Statistics*, 1273–1282. PMLR.

McMahan, H. B., Moore, E., Ramage, D., and Arcas, B. A. (2016). Federated learning of deep networks using model averaging. *arXiv Preprint arXiv:1602.05629*, *2*, 2.

Mehta, S. (2017). Concept drift in streaming data classification: Algorithms, platforms and issues. *Procedia Computer Science*, *122*, 804–811.

Mishra, S., Bordin, C., Taharaguchi, K., and Palu, I. (2020). Comparison of deep learning models for multivariate prediction of time series wind power generation and temperature. *Energy Reports*, *6*, 273–286.

Mocanu, E., Nguyen, P. H., Gibescu, M., and Kling, W. L. (2016). Deep learning for estimating building energy consumption. *Sustainable Energy, Grids and Networks*, *6*, 91–99.

Moishin, M., Deo, R. C., Prasad, R., Raj, N., and Abdulla, S. (2021). Designing deep-based learning flood forecast model with ConvLSTM hybrid algorithm. *IEEE Access*, *9*, 50982–50993.

Munir, A., Kansakar, P., and Khan, S. U. (2017). IFCIoT: Integrated Fog Cloud IoT: A novel architectural paradigm for the future Internet of Things. *IEEE Consumer Electronics Magazine*, *6*(3), 74–82.

Naik, S., Verma, A., Patil, S. A., and Hingmire, A. (2021). Flood Prediction using Logistic Regression for Kerala State. *International Journal of Engineering Research*, *9*(3).

Nasiri, H., Nasehi, S., and Goudarzi, M. (2019). Evaluation of distributed stream processing frameworks for IoT applications in Smart Cities. *Journal of Big Data*, *6*, 1–24.

Nie, H., Liu, G., Liu, X., and Wang, Y. (2012). Hybrid of ARIMA and SVMs for short-term load forecasting. *Energy Procedia*, *16*, 1455–1460.

Pandya, S., Srivastava, G., Jhaveri, R., Babu, M. R., Bhattacharya, S., Maddikunta, P. K. R., and Gadekallu, T. R. (2023). Federated learning for smart cities: A comprehensive survey. *Sustainable Energy Technologies and Assessments*, *55*, 102987.

Pathak, A. R., Pandey, M., and Rautaray, S. (2020). Adaptive framework for deep learning based dynamic and temporal topic modeling from big data. *Recent Patents on Engineering*, *14*(3), 394–402.

Peng, Y., Wang, Y., Lu, X., Li, H., Shi, D., Wang, Z., and Li, J. (2019). Short-term Load Forecasting at Different Aggregation Levels with Predictability Analysis. *2019 IEEE Innovative Smart Grid Technologies - Asia (ISGT Asia)*, 3385–3390. https://doi.org/10.1109/ISGT-Asia.2019.8881343

Pentreath, N. (2015). *Machine learning with spark*. Packt Publishing Birmingham.

Piadeh, F., Behzadian, K., and Alani, A. M. (2022). A critical review of real-time modelling of flood forecasting in urban drainage systems. *Journal of Hydrology*, *607*, 127476.

Pokhrel, S. R., and Choi, J. (2020). Federated learning with blockchain for autonomous vehicles: Analysis and design challenges. *IEEE Transactions on Communications*, *68*(8), 4734–4746.

Pramanick, N., Acharyya, R., Mukherjee, S., Mukherjee, S., Pal, I., Mitra, D., and Mukhopadhyay, A. (2022). SAR based flood risk analysis: A case study Kerala flood 2018. *Advances in Space Research*, *69*(4), 1915–1929.

Savi, M., and Olivadese, F. (2021). Short-term energy consumption forecasting at the edge: A federated learning approach. *IEEE Access*, *9*, 95949–95969.

Saxena, D., and Singh, A. K. (2022). Auto-adaptive learning-based workload forecasting in dynamic cloud environment. *International Journal of Computers and Applications*, *44*(6), 541–551.

Sergei, K., Gornyy, V., Davidan, T., Latypov, I., Manvelova, A., Konstantinov, P., and Vasiliev, M. (2022). Satellite mapping of air temperature under polar night conditions. *Geo-Spatial Information Science*, *25*(2), 325–336.

Sharma, S. K., and Wang, X. (2017). Live data analytics with collaborative edge and cloud processing in wireless IoT networks. *IEEE Access*, *5*, 4621–4635.

Shi, H., Xu, M., and Li, R. (2017). Deep learning for household load forecasting—A novel pooling deep RNN. *IEEE Transactions on Smart Grid*, *9*(5), 5271–5280.

Shin, J.-Y., Kim, K. R., and Ha, J.-C. (2020). Seasonal forecasting of daily mean air temperatures using a coupled global climate model and machine learning algorithm for field-scale agricultural management. *Agricultural and Forest Meteorology*, *281*, 107858.

Sniderman, B., Mahto, M., and Cotteleer, M. (n.d.). *Industry 4.0 and manufacturing ecosystems, Exploring the world of connected enterprises, This document is available on the Internet*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Taïk, A., and Cherkaoui, S. (2020). Electrical load forecasting using edge computing and federated learning. *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, 1–6. IEEE.

Thorgeirsson, A. T., Scheubner, S., Fünfgeld, S., and Gauterin, F. (2021). Probabilistic prediction of energy demand and driving range for electric vehicles with federated learning. *IEEE Open Journal of Vehicular Technology*, *2*, 151–161.

Tran, D.-P., and Hoang, V.-D. (2019). Adaptive learning based on tracking and ReIdentifying objects using convolutional neural network. *Neural Processing Letters*, *50*(1), 263–282.

Wan, R., Mei, S., Wang, J., Liu, M., and Yang, F. (2019). Multivariate temporal convolutional network: A deep neural networks approach for multivariate time series forecasting. *Electronics*, *8*(8), 876.

Wang, L., Kubichek, R.,and Zhou, X. (2018). Adaptive learning based data-driven models for predicting hourly building energy use. *Energy and Buildings*, *159*, 454–461.

Webb, G. I., Lee, L. K., Petitjean, F., and Goethals, B. (2017, April 2). *Understanding Concept Drift*. arXiv. https://doi.org/10.48550/arXiv.1704.00362

Wu, X., Liang, Z., and Wang, J. (2020). Fedmed: A federated learning framework for language modeling. *Sensors*, *20*(14), 4048.

Xiao, F., Yang, M., Fan, H., Fan, G., and Al-Qaness, M. A. (2020). An improved deep learning model for predicting daily PM2. 5 concentration. *Scientific Reports*, *10*(1), 20988.

Yang, Q., Liu, Y., Chen, T., and Tong, Y. (2019). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *10*(2), 1–19.

Yang, T., Andrew, G., Eichner, H., Sun, H., Li, W., Kong, N., and Beaufays, F. (2018). Applied federated learning: Improving google keyboard query suggestions. *arXiv Preprint arXiv:1812.02903*.

Yang, Yang, Wang, Z., Zhao, S., and Wu, J. (2023). An integrated federated learning algorithm for short-term load forecasting. *Electric Power Systems Research*, *214*, 108830.

Yang, Yuting, Dong, J., Sun, X., Lima, E., Mu, Q., and Wang, X. (2017). A CFCC-LSTM model for sea surface temperature prediction. *IEEE Geoscience and Remote Sensing Letters*, *15*(2), 207–211.

Yildiz, B., Bilbao, J. I., Dore, J., and Sproul, A. B. (2017). Recent advances in the analysis of residential electricity consumption and applications of smart meter data. *Applied Energy*, *208*, 402–427. https://doi.org/10.1016/j.apenergy.2017.10.014

Yu, H., Sun, S., Yu, H., Chen, X., Shi, H., Huang, T. S., and Chen, T. (2020). Foal: Fast online adaptive learning for cardiac motion estimation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4313–4323.

Yu, X., Shi, S., Xu, L., Liu, Y., Miao, Q., and Sun, M. (2020). A novel method for sea surface temperature prediction based on deep learning. *Mathematical Problems in Engineering*, *2020*, 1–9.

Zhang, P., Wu, X., Wang, X., and Bi, S. (2015). Short-term load forecasting based on big data technologies. *CSEE Journal of Power and Energy Systems*, *1*(3), 59–67. https://doi.org/10.17775/CSEEJPES.2015.00036

Zhang, Q., Wang, H., Dong, J., Zhong, and., & Sun, X. (2017). Prediction of sea surface temperature using long short-term memory. *IEEE Geoscience and Remote Sensing Letters*, *14*(10), 1745–1749.

Zhang, Z., Qiu, J., Huang, X., Cai, Z., Zhu, L., and Dai, W. (2021). Comparing and Evaluating Macao Flood Prediction Models. *IOP Conference Series: Earth and Environmental Science*, *769*(2), 022001. IOP Publishing.

Zliobaite, I., Bifet, A., Gaber, M., Gabrys, B., Gama, J., Minku, L., and Musial, K. (2012). Next challenges for adaptive learning systems. *ACM SIGKDD Explorations Newsletter*, *14*(1), 48–55.

Zliobaite, I., and Gabrys, B. (2012). Adaptive preprocessing for streaming data. *IEEE Transactions on Knowledge and Data Engineering*, *26*(2), 309–321.

Žliobaitė, I., Pechenizkiy, M., and Gama, J. (2016). An Overview of Concept Drift Applications. In N. Japkowicz & J. Stefanowski (Eds.), *Big Data Analysis: New Algorithms for a New Society* (pp. 91–114). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-26989-4_4

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : ABDULLA, Nawaf
Uyruğu :
Doğum tarihi ve yeri :
Medeni hali :
e-mail :

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Doktora	Gazi Üniversitesi / Bilişim Sistemleri	Devam ediyor
Yüksek lisans	JUST / Bilgisayar Bilimleri	2014
Lisans	Aden Üniversitesi / Bilgisayar Mühendisliği	2007
Lise	Lutfi Aman / Bilim Lisesi	2001

İş Deneyimi

Yıl	Yer	Görev
2022-Halen	Energy Pool – İstanbul	Veri Bilimcisi
2018-2020	Gazi Üniversitesi - TÜBİTAK	Proje Asistanı
2007-2017	Aden Üniversitesi	Araştırma Görevlisi

Yabancı Dil

Arapça (ana dili),
İngilizce,
Türkçe,
Fransızca (temel)

Yayınlar

- Abdulla, Nawaf, Mehmet Demirci, and Suat Özdemir. "Adaptive learning on fog-cloud collaborative architecture for stream data processing." In 2021 International Symposium on Networks, Computers and Communications (ISNCC), pp. 1-6. IEEE, 2021.

- Abdulla, Nawaf, Mehmet Demirci, and Suat Ozdemir. "Design and evaluation of adaptive deep learning models for weather forecasting." Engineering Applications of Artificial Intelligence 116, 2022: 105440.

- Abdulla, Nawaf, Mehmet Demirci, and Suat Ozdemir. "Towards Utilizing Unlabeled Data for Flood Forecasting with Weakly Supervised Adaptive Learning." In 2023 Innovations in Intelligent Systems and Applications Conference (ASYU), pp. 1-6. IEEE, 2023.

- Abdulla, Nawaf, Mehmet Demirci, and Suat Ozdemir. " Smart Meter-based Energy Consumption Forecasting for Smart Cities using Adaptive Federated Learning." In Sustainable Energy, Grids and Networks (*submitted in May 2023, accepted with minor revisions*).

- Abdulla, Nawaf, Sedef Demirci, Mehmet Demirci, and Suat Ozdemir. " Chapter: Adaptive learning in IoT-based smart city applications." In Principles and Applications of Adaptive Artificial Intelligence Book (*accepted, in production*).



GAZİLİ OLMAK AYRICALIKTIR.