TU WIEN Informatics

Towards Livestock Location Prediction in Rural Areas using Federated Learning and Edge Computation

MASTERARBEIT

zur Erlangung des akademischen Grades

Master of Science

im Rahmen des Studiums

Data Science

eingereicht von

Marko Gugleta, BSc
Matrikelnummer 12016483

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof.in Mag.a rer.soc.oec. Dr.in rer.soc.oec. Ivona Brandic
Mitwirkung: Univ.Ass. Alessandro Tundo, PhD
Projektass. Ahmad Sabtain, MSc

Wien, 2. Mai 2025

Marko Gugleta	Ivona Brandic

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

TU WIEN Informatics

Towards Livestock Location Prediction in Rural Areas using Federated Learning and Edge Computation

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Data Science

by

Marko Gugleta, BSc
Registration Number 12016483

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof.in Mag.a rer.soc.oec. Dr.in rer.soc.oec. Ivona Brandic
Assistance: Univ.Ass. Alessandro Tundo, PhD
Projektass. Ahmad Sabtain, MSc

Vienna, May 2, 2025

Marko Gugleta | Ivona Brandic

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Erklärung zur Verfassung der Arbeit

Marko Gugleta, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.
Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, haben ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT- Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 2. Mai 2025

Marko Gugleta

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Danksagung

Ich danke meinen Vorgesetzten für ihre kontinuierliche Anleitung und meiner Familie für ihre unerschütterliche Unterstützung.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Acknowledgements

Thank you to my supervisors for their continued guidance and and thanks to my family for their unwavering support.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Kurzfassung

Aufgrund der zunehmenden Technisierung der Landwirtschaft gewinnt die Viehhaltung zunehmend an Bedeutung, und neue Methoden entwickeln sich rasant. In ländlichen Gebieten gestaltet sich die Viehverfolgung jedoch aufgrund von Herausforderungen wie eingeschränkter Konnektivität und fehlender zentraler Speicherkapazitäten äußerst schwierig.

Die fehlende Möglichkeit, große Datenmengen konsistent zu übertragen, erschwert die Nutzung zentraler Datenspeicher und erschwert so das effektive Trainieren und Verteilen von Modellen. Darüber hinaus erschwert der Mangel an leistungsstarken zentralisierten Verarbeitungslösungen das Trainieren und die Inferenz von Modellen zusätzlich.

Bestehende Methoden zur Viehverfolgung nutzen meist nicht-maschinelle Lernmodelle oder zentralisierte Deep-Learning-Lösungen. Beide Methoden weisen jedoch unterschiedliche Nachteile auf. Nicht-maschinelle Lernmethoden, wie z. B. Modelle der ARIMA-Familie, haben Schwierigkeiten, komplexere und dynamischere Bewegungsmuster der Tiere zu erfassen. Zentralisierte Deep-Learning-Methoden hingegen sind zwar deutlich leistungsfähiger, benötigen aber große Datensätze und deren Übertragung auf einen zentralen Server. Dies stellt in ländlichen Gebieten aufgrund zahlreicher Einschränkungen, wie z. B. Bandbreitenbeschränkungen, eine erhebliche Herausforderung dar.

Um diese Einschränkungen zu umgehen, schlägt diese Arbeit einen Ansatz vor, der Federated Learning (FL) in Kombination mit Edge Computing zur Vorhersage von Viehpositionen in ländlichen Gebieten nutzt. FL ermöglicht es mehreren lokalen Geräten (Clients), ein gemeinsames globales Modell gemeinsam zu trainieren, ohne Daten an einen zentralen Server senden zu müssen. Anstatt Daten zu sammeln, werden lediglich Modellgewichte geteilt, was die Kommunikationskosten deutlich reduziert.

Empirische Ergebnisse zeigen, dass der FL-basierte Ansatz leistungsfähiger ist. Der Ansatz erreichte nach nur fünf Runden Server-Client-Kommunikation eine durchschnittliche Genauigkeit der Positionsvorhersage von ca. 52 Metern. Im Vergleich dazu zeigt dies eine signifikante Verbesserung der Genauigkeit um fast 50% gegenüber dem besten Nicht-ML-Ansatz, ARIMAX (101 m), und eine zusätzliche Steigerung von über 10% im Vergleich zu einem zentralisierten Deep-Learning-Ansatz. Die Ergebnisse zeigen, dass FL hochpräzise Vorhersagen liefert, ohne dass eine zentrale Speicherung erforderlich ist, und das Problem der eingeschränkten Konnektivität löst.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Abstract

Due to increasing presence of technology in agriculture, livestock management has become increasingly important and new methods are developing rapidly. However, rural areas make it extremely hard to track livestock and due to challenges such as limited connectivity and lack of centralized storage.

The inability to transfer large volumes of data in a consistent manner makes the usage of a centralized data storage highly unfeasible, making it hard to effectively train models and distribute them. In addition to this, the lack of powerful centralized processing solutions further complicates the training and inference of models.

Existing solutions of livestock tracking methodologies mostly use non-machine learning models or centralized deep learning solutions. The issue with those methods is that they both have different drawbacks. Non-machine learning methods, such as ARIMA-family models, have difficulty capturing some more complex and dynamic pattern movements of the livestock. As for the centralized deep learning method, in spite of being much more powerful, have a need for large datasets and their transfer to a central server. This creates a significant challenge in rural areas due to many constraints such as bandwidth limitations.

In order to get by these limitations, this thesis proposes an approach utilizing Federated Learning (FL) in combination with Edge Computing for rural livestock position prediction. FL gives the ability to multiple local devices (clients) to train a shared global model in collaboration without a need to send data to a centralized server. Instead of collecting data, only model weights are shared, which as a result significantly reduces communication costs.

Empirically the results show that the performance of the FL-based approach is superior. The approach achieved an average position prediction accuracy of approximately **52 meters** after only five rounds of server-client communication. In comparison, this shows a significant improvement with accuracy **increasing almost 50%** from the best non-ML approach, ARIMAX (101m), and and **additional increase of over 10%** in comparison to a centralized deep learning approach. The results show that FL is able to give high accuracy predictions while eliminating the need for centralized storage and addressing the problem of limited connectivity.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Contents

Kurzfassung **xi**

Abstract **xiii**

Contents **xv**

1 Introduction **1**

2 Background and Related Work **5**
2.1 Background 5
2.2 Approaches to Location Prediction 5
2.3 Computing on the Edge using Federated Learning 16

3 Federated Learning using an LSTM Model **19**
3.1 System Architecture 20
3.2 Mathematical Formulation 22
3.3 Key Challenges 25
3.4 Proposed Innovations 25
3.5 Advantages of the approach 26

4 Experimental Setup **27**
4.1 Data Characteristics 27
4.2 Data Preparation 31
4.3 Evaluation Metrics 36
4.4 Experiment Setup and Design 40
4.5 Implementation Considerations 40

5 Results **43**
5.1 Full dataset 43
5.2 Per device subdataset 47
5.3 Day/Night subdataset 50
5.4 Discussion and Limitations 52

6 Conclusion and Future Work **55**

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Overview of Generative AI Tools Used **57**

List of Figures **59**

List of Tables **61**

List of Algorithms **63**

Bibliography **65**

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 1

Introduction

With the increasing demand for precision agriculture and livestock management, various technological solutions have been developed to meet this need. These include technologies such as the Internet of Things (IoT), Distributed Machine Learning and edge computing. These technologies enable smart livestock monitoring by using sensors and machine learning prediction models. However, there are quite a few challenges that occur in rural areas such as limited internet connectivity, lack of centralized storage, as well as privacy concerns related to livestock data [Zhe15b]. In addition to this, edge computing directly addresses these issues by processing the data locally, which as a result reduces bandwidth usage and the latency. This decentralized nature of edge computing gives an ideal foundation for a more advanced approach to machine learning, such as Federated Learning [SCZ+16a].

This thesis contributes in the field of location prediction, specifically in the remote areas of the Alpine region. Most of the research has focused on urban areas for location prediction, where signal and connectivity issues are not a problem. The data accumulated is much more consistent and less prone to outliers, which reduces the difficulty of the mentioned task.

Additionally, the prediction for urban settings is much more predictable in comparison to prediction of livestock movement, since the livestock is freely grazing without any disturbances or limitation. For that reason, this thesis researches a different part of location prediction realm and explains how to use it in such conditions.

Federated Learning (FL) addresses the existing issue of limited connectivity by allowing decentralized model training, where only model updates are shared and the data from devices remains locally stored [YLCT19a]. Although, the communication overhead of Federated Learning can still be considerable in low-bandwidth environments, making efficient communication protocols and compression techniques necessary [KMY+16].

In spite of these advantages, deploying Federated Learning in rural areas for livestock location prediction remains challenging due to resource constraints, dynamic environmental conditions and the need for real time decision making [KYT+20]. For example, limited connectivity and device heterogeneity, can make the communication difficult and cause model updates to fail. In addition to this, dynamic environmental changes can corrupt sensor data and cause the models to become less accurate over time.

This thesis aims to address the aforementioned challenges by developing a framework using Federated Learning and Edge Computing for livestock prediction in rural areas. Additionally, this thesis will contribute to research carried out in the light of the FFG Virual Shepherd project which goal is the use of integrated satellite-based and terrestrial communication and positioning methods as well as Geo-spatial Artificial Intelligence & High Performance Computing for sustainable, safe and economical grazing in the Alpine region [FFG24]. The mentioned framework will be developed with the focus on optimization of model accuracy, while reducing communication overhead and ensuring energy efficiency, additionally maintaining data privacy and adapting to the dynamic nature of rural environments. The research will also explore the trade-offs between model complexity, prediction accuracy, and resource usage in resource-constrained settings.

The results from this thesis confirm that a FL approach can significantly solve the challenges which are normally present in the traditional methods. The FL model shows good performance, comparable to that of centralized approaches, in addition to an advantage of lower energy consumption and execution time, demonstrating that FL is a more efficient solutions for edge devices.

In the first chapter, the problem of the thesis and the motivation behind it is explained. After the introductory chapter, background and related work is investigated in order to see where current research is and where this thesis fits.

Since this thesis doesn't have a similar enough counterpart to be able to compare the data, a baseline needs to be established in order to determine how effective the more advanced methods are. After, the more complex method using deep learning will be explained.

Next, concept of federated learning is explained in chapter 3, which outlines why it has good application for this region where connectivity is limited. In addition to the explanation of advantages, the core method od server and client communication is explained and a pseudo code algorithm is covered.

In chapter 4, experimental setup is explained which mentions the data characteristics and how it is prepared to be used to build prediction models. In addition to this, evaluation metrics are given which will help determine how successfully and effective trained models are, and whether there is a certain trade-off between them. Experiment Design is then meticulously outlined, in order to make sure no details in the setup is missing and could lead to incorrectness. Lastly, some implementation details are mentioned to ensure reproducibility.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Chapter 5 then covers the results and gives a discussion comparing them and determining which approach was the most effective and performed best when looking at multiple different factors. Lastly, chapter 6 covers the final discussion and the conclusion and also gives some insight on potential future work that could be done on this topic.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 2

Background and Related Work

2.1 Background

Location prediction has become a critical research area with its many application including urban computing, transportation systems and different location-based services [Zhe15b][LWL+16]. Some recent advancements made using deep learning and spatial-temporal data analysis have significantly improved the accuracy and efficiency of these tasks [LWL+16].

Location prediction is a process of forecasting future position based on the historical data of movement from a certain object or person. In this scenario, the goal is to forecast future position of livestock, using the previous recorded location. Since the data is scattered and the signal is weak, the best way to build a model and make a prediction is to use a method where the model can be trained with less data transfer. For this reason, Federated learning in combination with edge computing is the perfect use case for this scenario and the topic of the thesis.

This chapter investigates the current state of the art methods in trajectory prediction, spatial-temporal modeling, but also federated learning and edge computing in addition, highlighting different key methodologies and challenges.

2.2 Approaches to Location Prediction

Location prediction can be split into multiple categories, such as trajectory prediction and spatial-temporal predictions. Trajectory prediction has the aim to forecast the possible future locations of moving objects that are based on their previous trajectories. The traditional methods usually rely on some type of probabilistic models, such as Markov chains and Hidden Markov Models (HMMs), to encapsulate movement patterns [Zhe15a].

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

In spite of the methods working well on smaller data, they struggle with some long-term dependencies and more complex spatial-temporal correlations [Zhe15a]. Another traditional method is family of ARIMA models, which are a probabilistic approach where the goal is to forecasting.

2.2.1 Non-Machine Learning Approaches

Autoregressive Integrated Moving Average (ARIMA), with its wide extensions, namely ARIMAX (ARIMA with the addition of exogenous variables) and SARIMAX (Seasonal ARIMA with the addition of exogenous variables), are widely popular because of their flexibility in capturing temporal dependencies when forecasting. This chapter provides a comprehensive overview of these models, with the addition of their mathematical formulations, the application in location prediction as well as some advantages and disadvantages.

ARIMA Model

ARIMA model, introduced in 1970 by [BJ70], is a model that is fitted with timeseries datasets. It is used to better understand the data and to make some predictions about possible future values. It generalizes the data in order to make the predictions as good as possible.

The model consists of several different parts, which all serve different function in order to make sure the completed model works well considering all factors.

Components of ARIMA

The ARIMA(p, d, q) model consists of:

- AR(p): Autoregressive component of order p. Autoregressive would mean that a model is able to predict possible future values that are based on past values. Parameter p describes the number of lagged values in the model.
- I(d): Integration component of order d. By differencing the data, this part handles the non-stationarity of the time series. The parameter d describes the number of times the data is differenced before achieving stationarity.
- MA(q): Moving average component of order q. This parts handle the modeling of the curret value of the time series as a linear combination of the past forecast errors. The parameter q describes the number of lagged error terms.

The orders of each component (p, d, q) could be considered as hyperparameters that are later useful for model optimization in the sense of making the model generalize better.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Mathematical Formulation

In order to understand a bit better how the ARIMA model works, the mathematical formulation is shown for time series data. If there is a current time t, the value for that time is y_t. In order to show how the calculation of y_t is done, there are some terms that need to be explained to properly grasp the formulas.

The first term is the lag operator L. It helps with the notational convenience by representing the previous or future time points ($L^k y_t = y_{t-k}$). The second term is the while noise error term ϵ_t, which is used to represent the represent random changes in the data. It is drawn from the normal distribution and calculated as $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$.

Now, in order to formulate the whole ARIMA(p, d, q) model, the definition of the Autoregressive(AR) and the Moving Average(MA) are necessary. The Autoregressive part can be defined as following:

$$\epsilon_t = \Phi(L) y_t \tag{2.1}$$

where:

- $\phi_1, \phi_2, \ldots, \phi_p$ are the autoregressive parameters that are trying to determine the influence of past values on the current one.
- $y_{t-1}, y_{t-2}, \ldots, y_{t-p}$ are the previous values in the time series.
- $\Phi(L) = 1 - \phi_1 L - \phi_2 L^2 - \cdots - \phi_p L^p$ is the autoregressive polynomial.

Now for the Moving Average (MA), the formula is:

$$y_t = \Theta(L) \epsilon_t \tag{2.2}$$

where:

- $\theta_1, \theta_2, \ldots, \theta_p$ are the moving average parameters.
- $y_{t-1}, y_{t-2}, \ldots, y_{t-p}$ are the previous values in the time series.
- $\Theta(L) = 1 + \theta_1 L + \theta_2 L^2 + \cdots + \theta_q L^q$ is the moving average polynomial.

Since both necessary components are well defined, the full formula can be explained. For y_t, the ARIMA(p, d, q) model is formulated as following:

$$\Phi(L)(1 - L)^d y_t = \Theta(L) \epsilon_t \tag{2.3}$$

where:

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

- L is the lag operator.
- d is the differencing parameter.

In this formula $(1 - L)^d$ is the Integration part (I). It differences the data d times. Here is an example for different values:

- for $d = 1 : (1 - L)^1 y_t = y_t - By_t = y_t - y_{t-1}$. This repesents the first order difference.
- for $d = 2 : (1 - L)^2 y_t = (1 - 2L + L^2)y_t = y_t - 2Ly_t + L^2 y_t = y_t - 2y_{t-1} + y_{t-2}$. This represents the second order difference.

With this, every part of ARIMA is combined in order to get the best predictions for timeseries data.

ARIMAX Model

As an extension for ARIMA, ARIMAX tries to incorporate different exogenous variables in order to get even better predictions. Exogenous variables are independent variables that most likely have an influence on the forecasting of the dependent variable, also called endogenous variable.

The goal is to determine the relationship between them and to leverage it into making better predictions, and there can be one or more exogenous variables incorporated in the model.

Extension with Exogenous Variables

The exogenous variables can be represented in the following way: $(X_{1,t}, X_{2,t}, \ldots, X_{m,t})$ for m different variables. In order to include them in the formula, they can be represented as the following part of the equation:

$$\beta_i(L)X_{1,t} + \beta_i(L)X_{2,t} + \cdots + \beta_m(L)X_{m,t} \tag{2.4}$$

where:

- $\beta_i(L) = \beta_{i,t} + L\beta_{i,1} + L^2\beta_{i,2} + \cdots + L^n\beta_{i,n}$ is the polynomial that represents the impact of the i-th exogenous variable.
- n is the number of lags that are included for each variable.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Now in order to get the formula for ARIMAX, the exogenous variables are incorporated into the formula from ARIMA. The updated equation is:

$$\Phi(L)(1-L)^d y_t = (\beta_i(L)X_{1,t} + \beta_i(L)X_{2,t} + \cdots + \beta_m(L)X_{m,t}) + \Theta(L)\epsilon_t \tag{2.5}$$

where:

- β_k is the coefficient for the i-th exogenous variable
- m is the number of exogenous variables

ARIMAX is mostly similar to ARIMA apart from the exogenous variables. But, ARIMAX has the potential to understand and predict data better with the help of additional information.

SARIMAX Model

In addition to the exogenous variables from ARIMAX, SARIMAX tries to incorporate the seasonal changes that are appearing in the time series data and tries to leverage them to make more accurate predictions. This is done by modeling the patterns that are happening in the data at regular intervals.

Seasonal Extension

Seasonality can be immediately be introduced into the full formula, extending from the previous ARIMAX formula. SARIMAX$(p,d,q)(P,D,Q)_s$ adds seasonal components:

$$\Phi(L)\Phi(L^s)(1-L^s)^D(1-L)^d y_t = (\beta_i(L)X_{1,t}+\beta_i(L)X_{2,t}+\cdots+\beta_m(L)X_{m,t})+\Theta(L)\Theta(L^s)\epsilon_t \tag{2.6}$$

where:

- s is the seasonal period
- D is the seasonal differencing order
- $(1-L^s)^D$ is the seasonal differencing operator
- $\Phi(L^s)$ is the seasonal autoregressive polynomial
- $\Theta(L^s)$ is the seasonal moving average polynomial

Entering seasonality into the equation gives another way the data can be more closely understood in order to give more precise predictions.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Usability in Location Prediction

Since predictions need to happen for both longitude and latitude, there needs to be two models for the predictions, since ARIMA-family models can only predict one feature at a time [HA21]. This would mean that for ARIMA model, the predictions for the features would have to be done separately, having no connections between them. This could effect the predictions, and that is why the use of exogenous variables is crucial. When using the ARIMAX model, the prediction can be done separately, but if the other feature is given as an exogenous variable, it will make much more sense to capture the influence one feature has over the other [BJRL15]. Lastly, since only a couple of months of data is used, SARIMAX might not be utilized to its full potential, since not a lot of seasonality is introduced into the data yet, making the ARIMAX most likely the best choice out of the three models [BD16].

Advantages and Limitations

ARIMA-family models are a simple way to make predictions for time series data. Through relationship in the data between different variables, patterns are established and the used to make the best predictions possible. The models are light weight and don't require too much data, making them an easy choice for time series data predictions. But, they can't completely capture nonlinear patters that some deep learning models do. In the next chapter in Table 2.2, the comparison between ARIMA-family models and the deep learning approach is given, in order to fully grasp the advantages and disadvantages each approach gives.

2.2.2 Machine Learning Approaches

Some more recent deep learning approaches try to address these limitations. **Recurrent Neural Networks (RNNs)**: RNNs, in particular Long Short-Term Memory (LSTM) neural networks have been widely adopted for trajectory prediction as a result of their ability to model sequential data. As an example, an LSTM-based model was proposed by Liu et al. [LWL+16] in order to predict the next location by capturing temporal dependencies in the trajectory data.

Attention Mechanisms: Even though attention mechanism was introduced for language modeling and machine translation [VSP+17], they have been implemented in many different type of models in order to improve their performance. This is why attention mechanisms have been integrated into LSTM models in order to improve prediction accuracy by giving additional focus on the most relevant parts of the input sequence. Liang et al. [LKZ+18] gave introduction to GeoMAN, a multi-level attention network that captures both spatial and temporal dependencies in geo-sensory time series.

Analysis of the spatial-temporal data always involves the understanding of the interplay between the spatial and the temporal dimension in the data [CW15]. This is especially significant for applications such as traffic flow prediction, urban mobility analysis and environmental monitoring. Some of the key advancements in this particular area include:

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Spatial-Temporal Neural Networks: Some deep learning models, such as Convolutional Neural Networks (CNNs) and Residual Neural Networks (RNNs) have been combined to capture the essence of both spatial and temporal features. Zhang et al. [ZZQ17] proposed a deep learning spatio-temporal residual neural network for citywide flow prediction, which integrates both spatial and temporal information using residual learning.

Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) have become a fundamental deep learning architecture for sequential data processing. Introduced by [HS97], LSTMs address the vanishing gradient problem through some of their sophisticated gating mechanisms, making them much more effective for time series forecasting problems where long-term dependencies are crucial.

LSTM is a special type of a recurrent neural network (RNN), which addresses the vanishing gradient problem, which were common in more traditional recurrent neural networks.

Recurrent neural networks are network which loop back from the output layer to the input layer, creating a connection which allows for the information to stay relevant. But, since those connections only occur from one step to the other, sometimes context is forgotten in between and not a lot of long term dependencies are able to be carried on, unabling the model to make a connection.

The way LSTM build up to the RNNs is that it remembers the information between dependencies for a longer period of time, thus getting its name Long Short-Term Memory. The most standard RNNs have only the most basic layers, while LSTMs consists of a special LSTM cell which enables this great advantage.

In the next section, the architecture of the LSTM cell is discussed and its relevant concepts, as well as the mathematical formulation of the LSTM unit.

Since LSTM was invented, it has seen many uses ranging from time series analysis, speech recognition, robot control, and even video games, and is widely popular in research as well with many different improvements.

LSTM Architecture

To fully understand the LSTM network architecture, core concepts are necessary to be understood to later understand the mathematical formulation. LSTM network layers are composed of LSTM cells and each cell has a state.

Each cell has two crucial components: **Hidden state (h_t)** and **Cell state (c_t)**. The hidden state is basically the same concept from the regular RNNs and it has a purpose of making the prediction at a current time step and has a task of being passed onto the next one.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

But, the cell state (c_t) is the core of the LSTM's principle and what makes it work so well. It is simply a vector which stores information, that runs through the cells in a sequence. It stores relevant information from any time step, which is how the vanishing gradient problem is mitigated. In Figure 2.1, the LSTM cell can be seen in a visual representation.

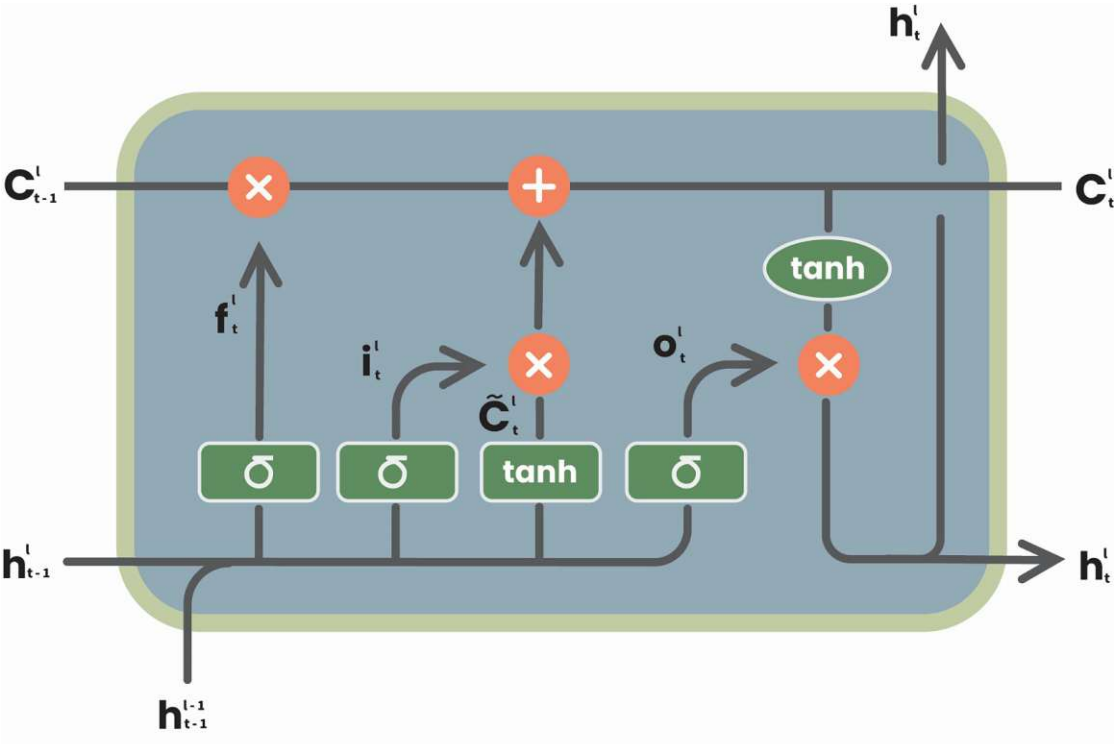

Figure 2.1: Visualization of LSTM cell architecture

In addition to the states, there are some input mechanisms which help regulate the information flow inside of a cell. The first one is the forget gate, which is used do decide whether the information from the previous cell state should be forgotten. The previous cell state goes in coordination with the current input though the sigmoid function, and dependent on the output, the information is either kept or completely lost.

The next one is the input gate, which is the deciding factor what new information will be stored in the cell state from the current input. The whole process has two different parts, the first one being the part where the updating parts are decided, and the second one being a creation of the update part. The decision of the update part selection falls again to the sigmoid function, and then an additional tanh function creates new information that will be added to the cell state.

The last gate is the output gate, which has a task to decide which data of the current cell state will be output in the hidden state. The way this gate work is that the sigmoid

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

function determines which parts of the cell state will be output, and then that determined cell state is given to a tanh function to output the values in the range between [-1, 1].

Since now the workings of the states and gates is known, in order to connect all of the principles described, the information flow is shown in order to grasp how the gates and the states are connected in the whole picture.

Mathematical Formulation and Information flow

Since the data in use here is timeseries data, every step is some time step t. During every time step t, there is some current input x_t, and a previous hidden state h_{t-1}.

What happens then are the gate activations, given in formulas:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.7}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.8}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.9}$$

where:

- f_t, i_t, o_t are the forget, input, and output gate activations
- W are weight matrices for each gate respectively
- b are bias vectors for each gate respectively
- σ is the sigmoid function

Now that the gates are activated, the cell state is updated in order to prepare it for the next time step. That can be represented in the following formulas:

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{2.10}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{2.11}$$

where

- C_t is the cell state at time t
- $\tilde{C}_t$ is the candidate cell state at time t

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

- $\odot$ denotes element-wise multiplication

As shown in formulas above, fist the candidate cell state is calculated from the hidden cell, and then in combination with the previous cell state and both forget and input gates, the new cell state is calculated. Lastly, the new hidden state is calculated as:

$$h_t = o_t \odot \tanh(C_t) \tag{2.12}$$

where:

- h_t is the hidden state at time t

After the new hidden state is generated, it is passed forward to the next time step in the combination with the cell state, creating a network of information.

Hyperparameters

In order to get the maximum out of a deep learning model, hyperparameter tuning is necessary to be done. It sets the hyperparameters in such a way to that the model training can be done in the most efficient way.

In Table 2.1, all hyperparameters that will be tested can be seen. The first hyperparameter, clipvalue, makes sure to clip the gradient of each weight to be no higher than this value. This makes sure that no gradient exploding occurs which could lead to problems when training a model.

The number of epochs dictates how many rounds of training will be conducted. It is an important value, since a little value would lead to weak generalization of data, and too many epochs would lead to overfitting which is not preferred.

The learning rate dictates how fast the model learns, i.e. how fast the model weights are updated. Fast learning does not always mean something positive, since the models needs to learn at a steady pace to be able to capture all dependencies in the data. Lastly, number of units determines how many LSTM cells does the layer have. This is an important hyperparameter as well, since it determines the size of the hidden units as well, which dictate how well the model can understand dependencies in the data.

Hyperparameter	Optional Values
clipvalue	[0.0, 1.0, 0.5]
epochs	[10, 20, 50]
learning_rate	[0.001, 0.0005]
units	[64, 128, 256]

Table 2.1: All Hyperparameters taken into consideration

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

The hyperparemeter tuning is done using a simple grid search, where the best loss determines the optimal hyperparameters. With that result, the models are later trained according to the Experiment process in Chapter 6.

Applications in Location Prediction

Since the data used here is a timeseries data, it is important to use a model which can efficiently capture the temporal dependencies and later present them in the location predictions. That is exactly why LSTM is the best option for a task like this.

Combining this model with the federated learning approach to model training could be the perfect combination tackle restrictions while delivering best results for the predictions.

Comparison with Traditional Methods

Now that both concepts of the ARIMA-family models and LSTM are known, it is important to discuss the differences they have and what advantages and disadvantages they bring to position prediction.

In Table 2.2, the comparison between the two methods can be clearly seen. It highlights some of the more important features, and outlines how each methods performs against them.

Feature	**ARIMA-family**	**LSTM**
Nonlinear patterns	Limited	Excellent
Long-term dependencies	Requires differencing	Native support
High-dimensional data	Challenging	Naturally handles
Training data required	Moderate	Large
Interpretability	High	Low

Table 2.2: Comparison of LSTM with ARIMA-family Models

There are clearly advantages and disadvantages to both methods, but it is just important to understand what they are and to get the most out of both methods.

Challenges and Limitations

In addition to the comparison with ARIMA-family methods, other limitations and challenges of LSTM need to be addressed. One of them is high sensitivity to hyperparameters. Even if a slight change occurs, a completely different result could be achieved, which increases the possibility to not get the full potential of the model, making hyperparameter tuning essential.

When working with edge devices, such as in this project, it is important to not have many complex operations in order to save as much as possible on energy and on execution

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

time. But when working with LSTM, a lot of computational complexity is introduced which could be problematic. That is why only one layer of LSTM will be used, in order to reduce the complexity and energy consumption as much as possible.

In spite of these challenges, LSTM offers much more advantages which will be helpful when solving the problem of location prediction.

2.3 Computing on the Edge using Federated Learning

A foundation and a more serious introduction to Federated Learning was given by [MMR+17], where the most commonly used algorithm, the Federated Averaging (*FedAvg*) was introduced. Since then it has become the most used algorithm when employing Federated Learning, and the go-to standard.

Federated Learning (FL) faces challenges such as system heterogeneity, characterized by diverse client characteristics, and statistical heterogeneity, where data is non-identically distributed (non-IID). To address these, [LSZ+20] introduced *FedProx*, a new FL algorithm. FedProx modifies and generalizes *FedAvg* through re-parameterization, leading to significantly improved results in heterogeneous environments and making it a more robust algorithm for such use cases.

Further research has focused on enhancing Federated Learning's already strong privacy properties. This includes the development of protocols for secure aggregation of high-dimensional data, such as the one developed by researchers at Google, which offers an additional option for privacy preservation in a federated setting [BIK+17a].

Edge Computing, introduced by [SCZ+16b], offers numerous advantages by addressing concerns such as response time requirements, battery life constraints, bandwidth cost savings, and enhanced data safety and privacy. The goal of Edge Computing is to move the computation and data storage as close to the source as possible. It enables reduction of latency and bandwidth usage. This paradigm also presents significant challenges, which continue to drive research in the field.

Even though Edge Computing is a secondary part of the Thesis, it is still an important topic to understand and also understand how it fits into the problem of Rural Area position prediction. Federated Learning moves both storage and computing to edge devices, i.e. closer to the source, and thus making the connection.

The inherent connection between Federated Learning and Edge Computing has led to the development of Edge Intelligence. Research comprehensively reviewed by [XYT+21] explores future opportunities in this combined domain.

Specifically within Mobile Edge Computing, the In-Edge AI framework was developed by [ZCL+19]. This framework intelligently utilizes collaboration among devices and edge nodes for exchanging learning parameters, thereby improving model training and inference.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Current research also focuses on integrating Trustworthy Artificial Intelligence (TAI) with federated learning, forming Trustworthy Federated Learning (TFL). This area aims to guide the development of more fair, robust, and explainable TFL systems [ZZL+24].

Overall edge computing is an important paradigm in this scenario since it offers many advantages that help blend in with the scenario of the thesis.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 3

Federated Learning using an LSTM Model

Federated Learning is a decentralized approach to model training which focuses on privacy preservation. Not only is it used for privacy protection, but for reduction of communication cost as well and it makes a centralized storage completely redundant. Introduced by Google [MMR+17] in 2017, it has become widespread due to its many advantages.

Federated Learning (FL) is a machine learning technique where multiple different devices, usually referred to as clients, train a model in a decentralized way, rather than following the usual paradigm and training it in a centralized manner.

While FL is mostly used for privacy concerns, this thesis has an unique approach that leverages the communication reduction and distributed training capability in order to address the critical challenge of livestock prediction, where the network connectivity is very limited. The data from individual sensor is scarce and isn't enough to train a model using only that data. FL allows to train a model from all of these sensors without having to collect and store that data centrally on the cloud which isn't even possible due to the extreme alpine environments. So instead of sharing/communicating raw data, only model updates (much smaller in size) are transferred to train a model. This thesis looks at how Federated Learning can achieve accurate location prediction while having minimal data transmission from edge devices located on livestock.

Since it could sound pretty similar to distributed learning, it is important to understand the difference between these two methods. Distributed learning has the goal to parallelize training, while federated learning has the goal to train on different data in a decentralized manner without sharing the details between clients.

The main goals are to cut the need to send data to one centralized place, making the whole procedure of training a model much more safer considering the privacy. Since the

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

data is not shared anymore, only model weights need to be sent to the server to train the model iteratively until a same or a similar result is achieved as in the centralized approach.

3.1 System Architecture

To explain the System Architecture in greater detail, it is important to define the different participants in the whole process. The participants are:

- Clients: Clients are the devices that run the models locally, and only send the updated weights, instead of all the data generated by that device. A client receives the current model weights from the server, performs a client weight update, and returns the newly updated weights to the server.
- Server: The server serves as a communicator between different clients. It receives the weight updates from clients, stores them in the generalized model, an performs different rounds of model training.

These participants described above can work in different architectures, the two most usual ones being the the use of central server and also peer-to-peer.

The centralized architecture requires both aforementioned participants, since it relies on the server to complete communication between clients. No communication occurs between clients at all, and only point of connection is the server. This makes the handling of the general model more easier since it is stored in one place. Even though the generalized data storage is not necessary since they are stored on the edge devices, the handling of the procedure of model training is much more easier. But, in case of no server available, peer-to-peer Federated Learning is at disposal.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

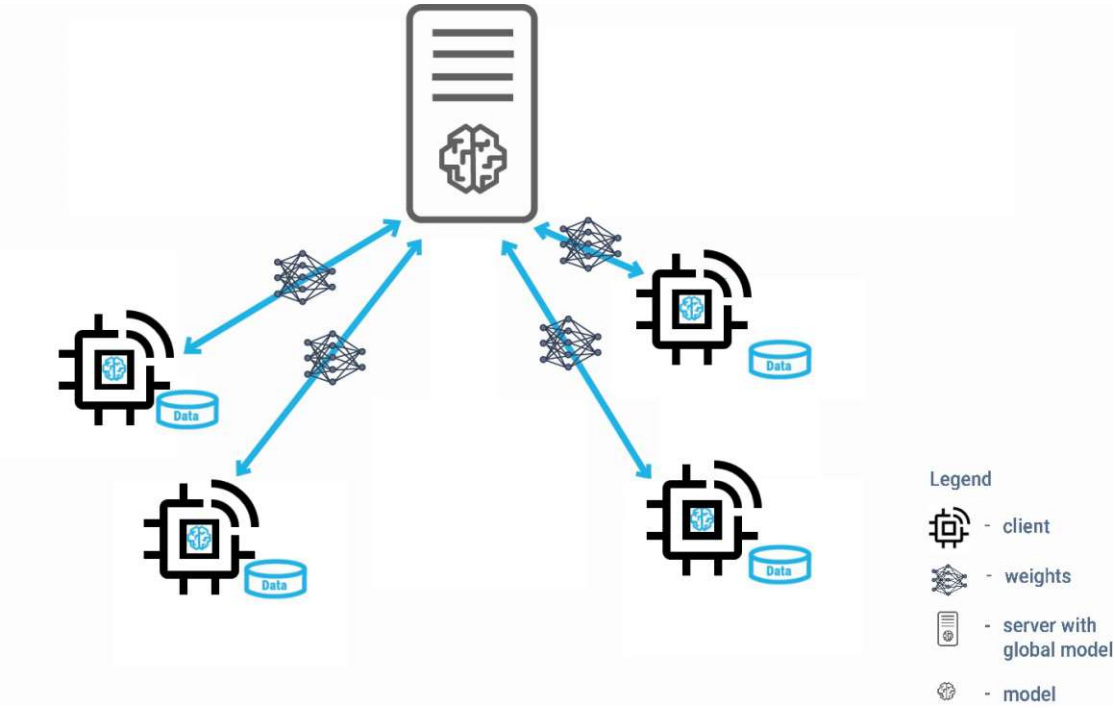

Figure 3.1: Visualization of the centralized architecture

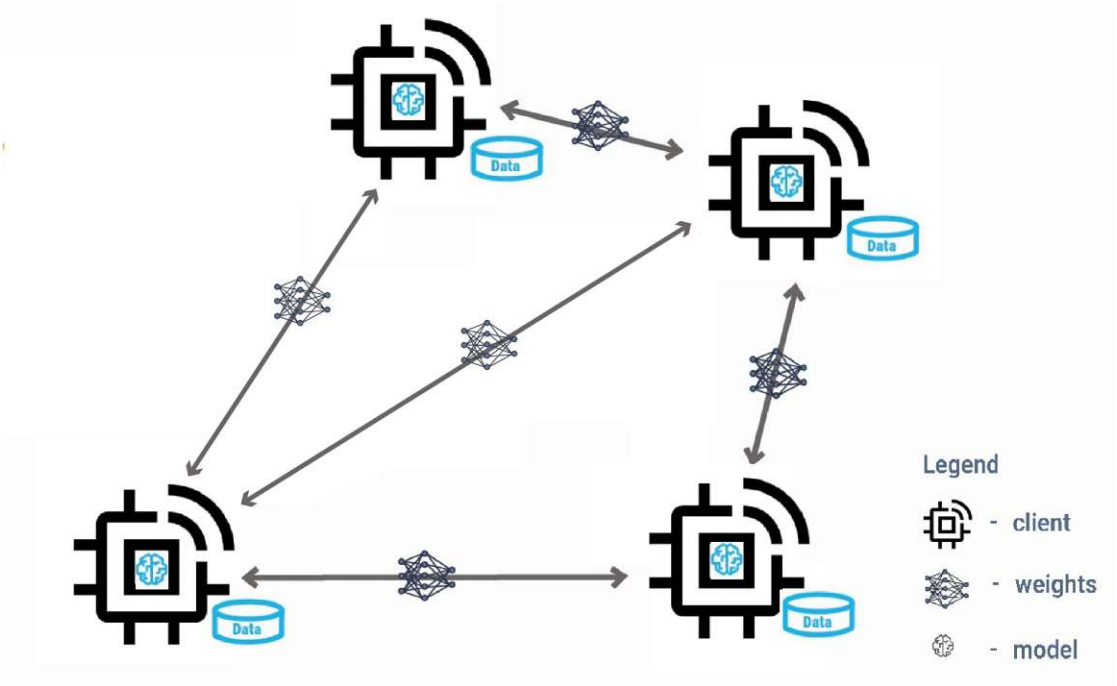

Figure 3.2: Visualization of the decentralized architecture

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

In this case, clients are the only participants in the process, and they communicate between each other. This approach, proposed by [CNS23], tries to mitigate some flaws in the original approach, such as data heterogeneity, partial device participation, and infrequent communication with the server.

In addition to the explanations, the are visualizations of both architectures available in Figure 3.1, where the centralized architecture is depicted, and Figure 3.2, where decentralized architecture is shown.

For the FFG Virtual Shepherd project, a centralized architecture (3.1) is more suitable for the task and will be used. Even though decentralized approach offers a robust and controlled environment, having a predefined set of livestock devices and a stable server infrastructure makes the centralized approach a better choice for aggregation and global model weights synchronization. In addition to this, the FFG project's goal of having the model later be used to make predictions available for the owners of the livestock aligns with the role of the server of the prediction distribution.

3.2 Mathematical Formulation

Now that the server architecture is known, it is important to mathematically explain the problem as well. The problem can be described as a minimization problem of a global objective function, which is simply an aggregation of multiple local loss functions from different clients.

The Federated Learning optimization problem can be expressed as:

$$\min_{\mathbf{w}} F(\mathbf{w}) = \sum_{k=1}^{K} \frac{n_k}{N} F_k(\mathbf{w}) \tag{3.1}$$

where:

- K is the total number of clients
- n_k is the number of samples at client k
- $N = \sum_{k=1}^{K} n_k$ is the total number of samples
- $F_k(\mathbf{w})$ is the local objective at client k

The objective is to find the global model weights that can minimize the following global objective function:

$$\mathbf{w}^* = \arg\min_{w} F(w) \tag{3.2}$$

Now that the goal of Federated Learning is clear from the mathematical perspective, the algorithm used for weight updates needs to be defined as well.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

3.2.1 Federated Averaging (FedAvg)

The most common algorithm for weight updates for Federated Learning is Federated Averaging, since it just averages the weights between all clients. This is also a method that will be used in this thesis for running the experiments. This algorithm is chosen exactly for this case, due to the specific application with an LSTM model for location prediction. The ability of the FedAvg model to average the model weights is very beneficial, since it allows the sequential nature of the LSTM's learned features to be combined across different clients and allows for all of the patters to be captured.

The global model updates the global weights $\mathbf{w}$ as:

$$\mathbf{w}^{(t+1)} = \sum_{k=1}^{K} \frac{n_k}{N} \mathbf{w}_k^{(t)} \tag{3.3}$$

where $\mathbf{w}_k^{(t)}$ is obtained by client k performing τ local optimization steps:

$$\mathbf{w}_k^{(t)} = \mathbf{w}^{(t)} - \eta \sum_{i=1}^{\tau} \nabla \ell(\mathbf{w}; \xi_i) \tag{3.4}$$

In addition to the formulas above, the algorithm for Federated Averaging is given in Algorithm 1, to further showcase how the principle of weights updates works.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Algorithm 1 Federated Averaging (FedAvg)

Require: Global model parameters $\mathbf{w}^{(0)}$
Require: Number of clients K
Require: Client participation fraction C
Require: Number of local epochs E
Require: Local batch size B
Require: Learning rate η
Require: Total rounds T

for round $t = 0$ **to** $T - 1$ **do**
  $S_t \leftarrow$ random subset of $\max(\lfloor C \cdot K \rfloor, 1)$ clients
  Broadcast $\mathbf{w}^{(t)}$ to all clients in S_t
  for each client $k \in S_t$ **in parallel do**
    $\mathbf{w}_k^{(t+1)} \leftarrow \text{ClientUpdate}(k, \mathbf{w}^{(t)})$
  end for
  Aggregate updates: $\mathbf{w}^{(t+1)} \leftarrow \sum_{k \in S_t} \frac{n_k}{n_{S_t}} \mathbf{w}_k^{(t+1)}$
    where $n_{S_t} = \sum_{k \in S_t} n_k$
end for
return final global model $\mathbf{w}^{(T)}$

ClientUpdate(k, $\mathbf{w}$)
$\mathcal{B}_k \leftarrow$ (split client k's data into batches of size B)
for local epoch $i = 1$ **to** E **do**
  for batch $b \in \mathcal{B}_k$ **do**
    $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla \ell(\mathbf{w}; b)$
  end for
end for
return $\mathbf{w}$ to server =0

The algorithm facilitates the training of a single global machine learning model across multiple decentralized clients. It operates in the following rounds:

- Server Sends: The central server broadcasts its current global model to a randomly selected subset of clients.
- Clients Train Locally: Each selected client trains this model independently on its own private, local dataset for a few epochs.
- Clients Send Updates: Clients send their updated model parameters (not raw data) back to the server.
- Server Aggregates: The server averages these received model updates, typically weighted by the size of each client's dataset, to create a new, improved global model.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

This iterative process leverages distributed computation to build a robust shared model.

3.3 Key Challenges

Even though Federated Learning is an approach which has many benefits, there are some challenges that it meets. This section explores and explains some of them.

3.3.1 Statistical Heterogeneity

- Non-independent and identically distributed (IID) data distribution across clients: Since the data comes from multiple devices, in an ideal case same number of points would come from the devices. In the case of this thesis, there is likelihood of highly localized data, due to natural movements of the livestock. Our approach will look into the impact of such data skew and how it influences the model convergence and its performance.
- Data skew: Since uneven data distribution is possible to happen, especially in data where anomalies could happen, data skewness could cause problem when building a prediction Federated Learning model. This is why removal of such data points is crucial to making data less skewed and more suitable for location prediction. Because of this, this thesis will also have a preprocessing step for all of the data in order to filter out the outliers and build a model on more stable data.

3.3.2 System Heterogeneity

- Network connectivity issues: Many issues could occur from bad connectivity, the main ones being failed or incomplete model updates, increased training time, or client dropout during training. These issues could cause a lot of problem when training a federated model, and is important to handle, Due to nature of the project, many of the devices will have weak connection with the server. But, using newly developed IoT protocols for wide range communication, this problem could be mildly mitigated.

3.4 Proposed Innovations

Here is a quick summary of all of the key contributions made in this thesis in the context of Federated Learning for location prediction in rural areas, in order to showcase the contributions and innovations given in this chapter and thesis:

- **Empirical Validation of FL for Communication Cost Reduction**: Minimization of communication overhead using Federated Learning's paradigm for real-time location prediction from edge devices on livestock, in a scenario where centralized data transfer and storage are much more difficult to execute regularly.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

- **Robustness Analysis of FedAvg with LSTM under System Constraints**: Combination of an LSTM model with Federated Learning in order to capture all natural movements of livestock in a single global model without sacrificing accuracy and performance.
- **Preprocessing for Non-IID Location Data**: A preprocessing pipeline of data with the goal of handling data anomalies and outliers contained in the movement data, directly enhancing model robustness and effectivity.

3.5 Advantages of the approach

The combined strength of Federated Learning and LSTM model provides a good solution that directly addresses the challenges that were identified. The capability of federated learning to average a model and combine it into a single global one is nicely complemented with LSTM's ability to predict sequences, erasing the need of multiple individual models, making it easy to maintain and scale. As such, the model can be applied to any number of cows on an already established server from the FFG Virtual Shepherd project. In addition to this, the privacy property of federated learning, which keeps the data on local devices, perfectly aligns with the paradigm of edge computing, making sure that communication costs are kept low and ensuring the movement patterns are never centrally exposed. This combination showcases a very robust and efficient way to approach and tackle this particular issue of location prediction in a remote environment.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 4

Experimental Setup

This chapter covers all necessary steps in the experimental setup which ensure correct production of results. Methods from chapters 2 and 3 will be used and tested in order to determine which method performs best. The setup consists of multiple steps, which consist of first and foremost data characteristics and its preparation. After that, different evaluation metrics are discussed.

4.1 Data Characteristics

This real-world dataset has been collected from actual livestock movement, since it is part of the FFG Virtual Shepherd project, which goal is the use of integrated satellite-based and terrestrial communication and positioning methods as well as Geo-spatial Artificial Intelligence & High Performance Computing for sustainable, safe and economical grazing in the Alpine region [FFG24].

The dataset is collected from collar sensors, which are attached to the livestock. The data is sent periodically, where the period is dependent of the time of the day. On average, every 20 minutes new information is sent from the sensor, increasing to at most 2 hours in the night, where the data is sent less frequently. This could lead to some problems when training the prediction models, and it will be addressed in the Data Preparation chapter.

The data consists of the features explained in Table 4.1. The features here are helpful for multiple different topics in the Virtual Shepherd project, but not all of them are suitable for position prediction. That is why only some of them are picked, and the reasoning is explained in Table 4.2.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Field	Type	Unit/Format	Description
altitude	numeric	meters	Height above sea level
battery	numeric	volts	Remaining battery voltage
customerID	string	-	Unique customer identifier
deviceEUI	string	hex	Extended Unique Identifier
deviceID	string	-	Device identification code
deviceName	string	-	Human-readable device name
deviceStatus	string	-	Current operational status
distance	numeric	meters	Measured distance
gatewayID	string	-	Gateway identifier
latitude	numeric	decimal degrees	GPS latitude coordinate
logtime	timestamp	ISO 8601	Time of log creation
longitude	numeric	decimal degrees	GPS longitude coordinate
paddock	string	-	Agricultural field identifier
rssi	numeric	dBm	Received signal strength
snr	numeric	dB	Signal-to-noise ratio
temperature	numeric	°C	Ambient temperature
time	timestamp	ISO 8601	Measurement timestamp

Table 4.1: Sensor Device Data

Feature	Rationale for Selection
time	Temporal patterns help identify movement trends and periodicity. Enables time-series analysis of trajectory data.
deviceID	Unique identifier for tracking individual devices. Essential for multi-device analysis and comparison. Allows device-specific behavior modeling.
latitude/longitude	Primary geospatial coordinates for position determination. Fundamental for mapping. Enable distance calculations between points.
altitude	Provides vertical position component. Helps distinguish positions with similar lat/long but different heights.

Table 4.2: Selected Features for Position Prediction

After the specific geo-spatial features were selected, it is important to explore the data to see what information needs to be further processed and which is already prepared. Overall, there are 21192 data points in the dataset, which are a combination of data from all devices. As for the number of devices, there are 8 of them in the dataset, where

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

the number of samples varies per device. In Table 4.3, number of samples per device is shown. Since one of the devices has just a single data point, it is omitted since that is not enough to make a decent prediction. Additionally, not all of the devices provided the same number of data points, which could lead to unequal representation in the later performance of the model. Although, there are enough points to represent all of the devices, which means each device will be represented enough in the model predictions later on.

Device ID	Value
1	1174
2	3950
3	1355
4	3912
5	4044
6	3911
7	2322
0	1

Table 4.3: Device Data

The geospatial data can be simply visually presented on a map, which can be seen in Figure 4.1. Each point represents latitude and longitude, where the legend represents when the data point was captured in time. As for where the location is more specifically, it is on the border between Austria and Germany, near the Königsee in Germany. The area where the livestock is tracked covers about 10km^2.

The area consist mostly of a combination of a forest and a meadow. That means that livestock has very little natural obstacles for movement, apart from trees. Since the altitude can't be represented on an image with the data points, it is shown additionally in Figure 4.2. The 3D representation shown nicely the altitude difference and how the livestock has to navigate the terrain.

There seems to be quite a lot of elevation differences which are noticeable in the grazing area of the livestock, after carefully examining Figure 4.2. This could be an additional obstacle for livestock movement, since it represents a natural border between different parts of the whole grazing area.

On the other hand, it could make the livestock movement much more predictable, since it could be more likely that the position won't change that much on the daily basis. Looking again at Figure 4.1, there are obvious two bigger clusters of points, with a gap between them. Again inspecting that same gap in Figure 4.2, there seems to be a significant altitude difference which prevents the livestock from crossing it.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

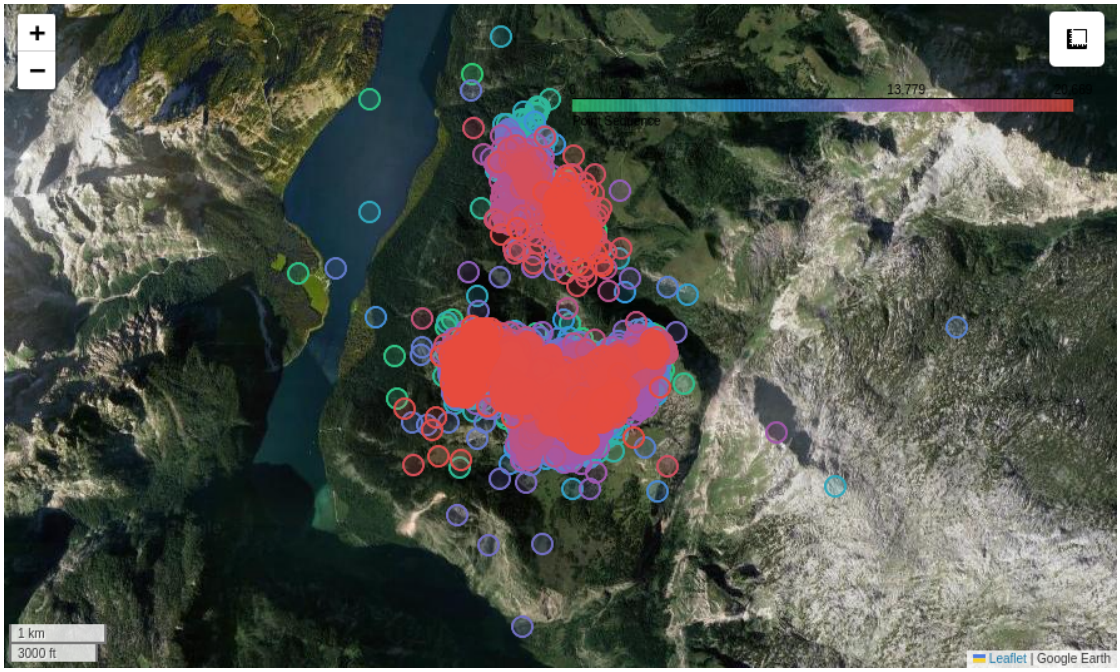

Figure 4.1: Visualized all data points of all of the livestock

Figure 4.2: Visualized grazing area to represent the altitude differences

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

4.2 Data Preparation

Data preparation will consist of multiple steps, in order to ensure that the data that will be used to train prediction models is properly processed. First, data formatting is done in order to bring the data into the preferred format, especially when working with timeseries data. Anomaly detection is the next natural step, since limited connectivity could lead to falsely reported data, which could possibly skew the prediction models. With the anomalies removed, data intervals would wary too much in terms of time interval difference, so data interpolation is necessary to make sure that time intervals are identical between all data points. The interval is set to 1 hour, so that the model can predict for up to two days, by predicting 48 future data points. After interpolation, one more round of anomaly detection is done in order to detect any possible anomalies left in the data.

4.2.1 Data Formatting

Fist step is to remove the unwanted features and only leave the ones necessary for position prediction. This means only latitude, longitude and altitude, with the addition of time stamps and device ID are used and the rest are removed.

Since there are no missing values, no data filling techniques needs to be applied. Just setting the correct data formats is necessary in order to later perform other preparation methods.

4.2.2 Anomaly detection

Due to weak connectivity, sometimes the reported livestock positions are not entirely correct. They could wary from being a few meters wrong to being few hundred meters wrong, which could significantly hinder the prediction models. As seen in Figure 4.1, there are single data points which are over the lake, which would not be physically possible for a single cow to traverse by itself. If kept in the dataset, these points could significantly skew the prediction models.

This is why it is important to check if and how many anomalies there are in order to ensure clean data. There are many ways how anomaly detection can be done, some being more complex and some being simple. Since this is not the focus of the thesis, the standard method for anomaly detection is done which is usage of z-score.

Data considered to calculate the z-score is divided per cow per day. That means only movement from one day and one cow is considered, which helps to see anomalies more clearly, since if more data was considered at the same time, the anomalies would become less obvious and blend in with the rest of the real data. There is an example of the data that is considered each day per device shown in Figure 4.3.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Figure 4.3: Visualized daily movement of one cow

Calculation of z-score

Based on the selected data, speed, acceleration and distance from mean are calculated as the metrics to then calculate the z-score. These metrics help to determine if the data points are considered anomalies, since they can be a good indicator for when the irregular movement is occurring.

Z-score is calculated using the following formula:

$$z_i = \frac{x_i - \mu}{\sigma} \quad \text{for } i = 1, 2, \ldots, n \tag{4.1}$$

where x_i is either speed, acceleration or distance from the mean on the daily basis (for one cow considering one day). Multiple z-scores are calculated for different metrics and then the average is taken to make sure that only real anomalies are detected.

If the z-score crosses a certain value threshold, in this case that value was set to 1, it will be detected as an anomaly, and then properly handled.

Manual handling of anomalies

Due to data being visually easy to interpret, looking at the data and understanding it is important since it can lead to discoveries of irregularities. Since the data includes locations of livestock from their farm, shown in Figure 4.4, it could easily lead to false predictions since the models will be used to predict positions in the forest, since the

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

distance from these two locations is around 10 kilometers. This is why manual removal from these data points is necessary, and it leads to removal of 2145 data points from the data set.

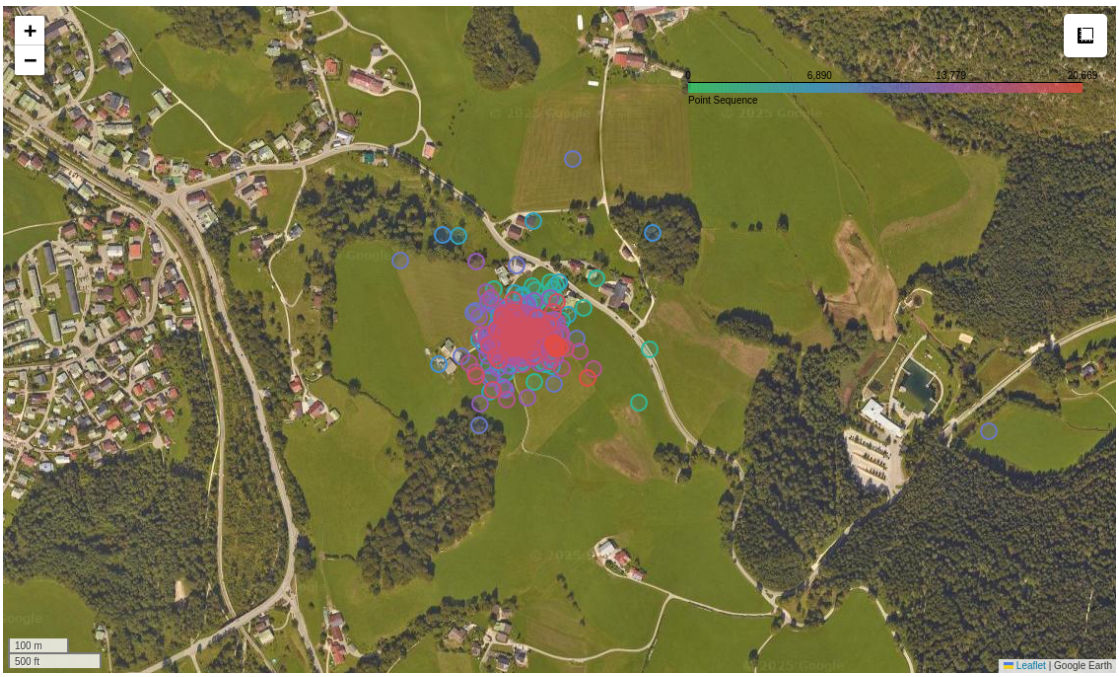

Figure 4.4: Visualized data points of cows in their farm

In addition to this, there is one point located in Styria, Austria. Since it has a distance of around 200 kilometers from all other data point, removing it also essential since it could move the mean for quite a significant amount.

Automatic Handling of anomalies

After concluding first round of anomalies using the technique described above, there are 2916 found out of 18647 data points left in the dataset. this presents a 15.63 percentage of the data, which is not insignificant. These points are simply removed from the data, since there are more than enough points in the dataset to build the prediction models. Since the current time interval average is around 20 minutes, and the desired interval is 1 hour, concluding that the removal of the selected data points is still beneficial.

4.2.3 Data Interpolation

Now that the first round of anomaly detection is done, data interpolation is performed in order to fill the gaps created by removed anomalies and to set the time interval to one hour. This is done using a pandas function *merge_asof*, which is similar to a left-join except that the nearest keys are matched rather than equal keys [pdt24]. It is performed

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

like this since not all daily time intervals are completely same, especially from different devices, creating a slight difference in the time stamps.

What is also important to mention is this interpolation is done per device since livestock movement could wary from animal to animal, because they are not strictly moving together. This leads to a creation of a new table with the new timestamps, where they are one hour apart from each other, to create equal timestamps. Then the existing data is sorted, since the function requires the data to be sorted to be able to work properly. After the two tables are ready, function is executed and the result is a table with the equally spaced timestamps, and geo-spatial data belonging to that time. This process can be seen visually in Figure 4.6.

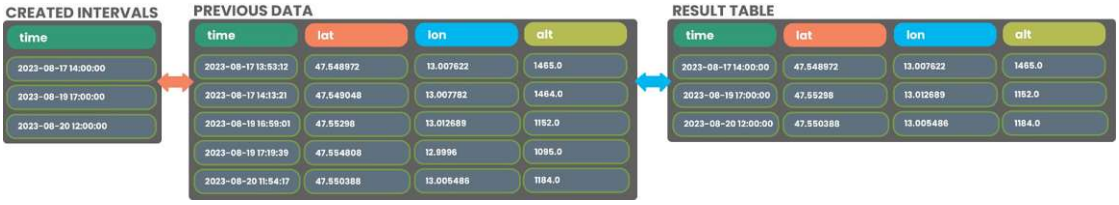

Figure 4.5: Visualized *merge_asof* function

After each devices' data is interpolated, they are merged with again with each ID from the devices before in order to keep the feature for later use. Before the whole interpolation process there were 15731 data points for all devices. After the process, since the time intervals have decreased, there are now 10099 data points across all devices.

4.2.4 Train/validation/test Split

As previously discussed, since models need to predict for up to 48 hours, and with the time interval set to one hour, 48 points need to be given for position prediction, in order to properly asses their performance.

This would mean that there only needs to be 48 data points in both validation and in test set. Since it is timeseries data, it needs to be properly sorted, from the first point in time to the last one.

After that a split is made, meaning everything but 96 points remains in the test set, where the number depends on the subdataset used.

4.2.5 Subdatasets

Since the dataset is a combination data from multiple devices, it is also important to consider different subdatasets that could be used for the prediction models. Of course the full dataset will be used as well, but looking at different approaches is important as well due to the nature of the dataset.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

In addition to the description of the subsdatasets, number of data points each one has will be given in a form of a table or text, to give a representation of how the data is split in terms of numbers as well.

Full data Subdataset

The full subdataset will use all the available data points in the dataset across all the devices. This means data has all of the movement patterns from all the cows. This could make it a bit harder for the model to predict, but it is much more practical since only one model is necessary to cover all the data.

The number of data points after all preparation steps are done is 9969. This does include the data from all devices, and the individual distribution will be seen in the following table.

Day/Night Subdataset

The Day/Night Subdataset is a variation of the full dataset where the data is split in two parts: the data collected during the day and the data collected during the night. Since the movement of animal does vary during the whole 24 hours of one day, it could lead to more accurate predictions depending on the time of the day. The Day data is from 6 in the morning to 6 in the night, whereas the rest of the data belongs to the night datasubset.

In Table 4.4, the number of points per device can be seen.

Time of Day	Data points
Day	4940
Night	5029

Table 4.4: Number of data points in each subset

Per device Subdataset

Per device Subdataset will use the data from each device individually in order to determine the future position of livestock individually. This could lead to better results than the general dataset, since the movements from each individual member of the livestock varies, and the models learns the movements of only one of them per model. Although, this does mean that more models are necessary to be trained, more specifically one per cow.

In Table 4.5, the number of points per device can be seen.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Device ID	Number of points
1	1915
2	1911
3	1911
4	1909
5	1090
6	655
7	578

Table 4.5: Number of points per device in the dataset

Now that the data is properly explored, prepared and divided other parts of the experiment design can be done. Next step would be to determine which metrics would be the most useful for the used dataset. After that, only experiment design and implementation details are discussed in this chapter.

4.3 Evaluation Metrics

In order to properly evaluate the models, it is necessary to use appropriate metrics. The most important one is probably the error metric, which evaluates the performance of the model on how far from the target value the prediction is. Another one is energy consumption, which tells how much electricity a model consumes during training and inference time. Lastly, execution time of again training and inference is evaluated, which is important due to limited resources.

4.3.1 Error metric

Since location prediction is a regression task, regression metrics need to be used, the most standard one being Mean Squared Error (MSE). But, due to the nature of the data, MSE is hard to apply in this scenario. It is best to shown this using an example. Taking two points in account, shown in Table 4.6, it is easy to calculate both metrics to compare them.

Point	**Latitude**	**Longitude**
Point 1	47.563776	12.992347
Point 2	47.562288	12.992175

Table 4.6: Coordinate Points

First, looking the the both points, they might seem almost the same. And looking at the Mean Squared Error from Table 6.5, it is exactly so. If this was used as an error metric, the deep learning model would not learn as much since the error seems so insignificant.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

But looking the the value of Haversine Distance from the same table, that error now seems much more significant, not only to the naked eye but the model as well.

Metric	Value	Unit
Haversine Distance	166.75	meters
Mean Squared Error	0.00000114	-

Table 4.7: Difference Between Two Coordinate Points

This is why it is important to use an appropriate error metric for the used data. As for why Haversine metric is more suitable, it is essential to know what it is and how it is calculated.

Haversine Distance Formula

The Haversine formula that is used here calculates the great-circle distance between two selected points on a sphere with their longitudes and latitudes. If there are two points, $L_1(lat_1, lon_1)$ and $L_2(lat_2, lon_2)$ in radians with their given coordinates, and R is the radius of the sphere (in our case R is the radius of the Earth). Then the formula of the Haversine distance is:

$$a = \sin^2\left(\frac{lat_2 - lat_1}{2}\right) + \cos(lat_1)\cos(lat_2)\sin^2\left(\frac{lon_2 - lon_1}{2}\right)$$

$$c = 2 \cdot \arctan 2(\sqrt{a}, \sqrt{1-a})$$

$$d = R \cdot c$$

where:

- $L_1 - lat_1, lon_1$: Latitude and longitude of the first point (in radians).
- $L_2 - lat_2, lon_2$: Latitude and longitude of the second point (in radians).
- R: Radius of the sphere.
- a: Intermediate value.
- c: Angular distance in radians.
- d: Haversine distance.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

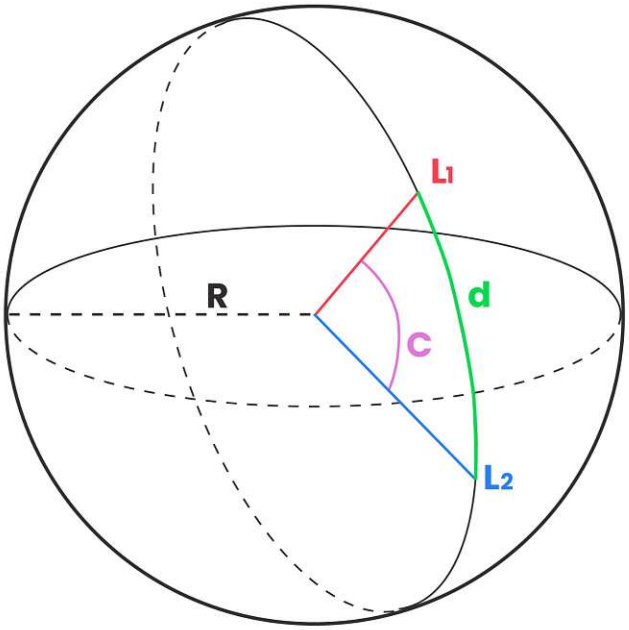

Figure 4.6: Haversine distance shown on the sphere

4.3.2 Model error metrics

For each model, there are three different metrics that show how well that model performs. The first one is the mean of error of the predicted points, where simply the average of the error of the predicted points is taken to showcase what the error is on the span of 2 days. Mean is calculated as following:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

The second one is the standard deviation, which represents the amount of variation of the values of the variable about its mean. In simpler terms, just the average of all the points from the mean. If the values are smaller, that means most of the points are near the mean, which could indicate that the predictions are more consistent in precision. Standard deviation is calculated as following:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \mu)^2}{n}}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

The last one is Root Squared Mean Error, which is an extension of the mean error, and it calculated like following:

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

The advantage of RMSE over the mean is that bigger errors contribute much more that smaller errors do. This gives a more accurate insight into the results and combined with the mean and the standard deviation, gives a much fuller picture when comparing performances.

4.3.3 Energy consumption

Due to multiple constraints, such as processing power of the devices the location is communicated from and also power they have, it is important to limit the energy consumption from the prediction models and have them use the least possible amount of electricity. That is why this metric is really important in addition to the error metric, since even if two models perform similarly, the one which consumes less electricity is prioritized due to the conditions.

One way to measure this metric is using the Running Average Power Limit (RAPL). This technology was developed by Intel and it estimates power consumption of a CPU [Cor23]. Its aim is to manage the power consumption withing specified limits over different time windows.

There is not a single, universal formula for RAPL, but rather it is a power management framework which is defined by configurable parameters and some internal processor logic. It is important to mention that all measurements will be done in Joules, so that the results will be comparable between different models.

4.3.4 Training and inference execution time

As the last metric, training and inference execution time will be used to determine how fast a model performs.

Again, due to the limited performance available, it is important for models to complete training and inference as soon as possible. Running a long time is not possible on an edge device such as these used in FFG Virtual Shepherd project, so choosing models with an acceptable runtime is crucial.

This is why this third and last metric is important to give a full picture of the model performance and help in the model evaluation.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

4.4 Experiment Setup and Design

The crucial question of this work is which subdataset is best suitable for the task of position prediction and using which model is the best result achieved. Also, whether federated learning model is able to perform as well as the other more conventionally trained models.

All model combinations will be performed with all subdatasets, in order to test all possible scenarios and to later compare every possible outcome.

Baselines will give a basic picture of how each subdataset compares to the other, and then the more advanced approach with deep learning will determine the best results based on the metrics discussed above.

Federated Learning is applied with the per device subdataset, since it can only be effectively used there, due to the nature of the technique. Still, this doesn't affect the other subdatasets since they are still relevant to see if any other methods could be a more efficient and better approach.

All of the experiments that were described above were implemented using Python version 3.9.21 and on Linux Mint 22.1 Cinnamon using 6.4.8 Cinnamon version. The machine has a AMD Ryzen 7 4800H with Radeon Graphics × 8 core processor paired with NVIDIA Corporation TU116M [GeForce GTX 1660 Ti Mobile] graphics card with an addition of 24 GB of RAM. The graphic cards takes the advantage of CUDA cores, using them to train deep learning models such as LSTM more efficient and faster.

As for hyperparameter optimization which is necessary for deep learning models, it was done by cross validation implemented from scratch. Only LSTM models from different data subsets need it, and it was done to improve the performance of them. Different models will be used for different techniques if necessary, since the goal should be to utilize the flexibility of the parameters for each different occasion, in this case different subsets that the models will be trained on.

For the models itself, baseline models were used from the Statsmodels, which come already fully implemented. The LSTM model is used from Tensorflow, and using their implementation the model was programmed layer by layer.

4.5 Implementation Considerations

When implementing the Federated Learning approach, there are some considerations that need to be mentioned such as which Global model update algorithm will be used and in which Framework the approach will be implemented.

- Global model update algorithm: The most common one will be used, which is Federated Averaging (FedAvg). This algorithm matches the goal perfectly, since every device should give an equal weight to the general model. And by averaging it, it makes it a perfect candidate for this particular use case.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

- Frameworks: There are many frameworks as well which allow for models to be trained in a federated manner, one of them being TensorFlow Federated. This would make perfect sense since it continues on the models already trained using TensorFlow. Even though the deep learning models will be implemented using TensorFlow, due to the flexibility of the Flower.io, it will be used over TensorFlow Federated. It gives a lot of flexibility when working with Tensorflow models and easy setup with the Federated Averaging update algorithm. As for the non ML-Approaches, they are implemented with the help of the statsmodels library. All three baseline approaches are there (ARIMA, ARIMAX and SARIMAX), with the availibility to change parameters when necessary. This gives a lot of flexibility when using these types of models and thus makes a great choice for this particular use case.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 5

Results

This chapter discusses results of the experiments that were ran as described in the previous chapter. The results cover all methods that were mentioned, including three baseline methods, a deep learning approach and a federated learning method.

The results are split based on the subdataset that they were ran on, meaning each subdataset has their own section where the results are shown. The metrics will be shown in Tables in order to clearly communicate the results from different methods. In addition to this, different Figures will be shown which give additional insight into the results such as the visualized prediction versus true value and distance change over time.

What is also important to mention for Federated Learning is that due to the nature of how the technique works, only results for the per device subdataset is available. As already discussed, since the system works with the data on devices and only the model weights shared, this is the only subset which works with Federated Learning. Why other methods are tested as well is to see whether other methods where Federated Learning is not used are viable to give good results in comparison.

Additionally, for every subdataset a hyperparameter tuning was done for the LSTM deep learning model, in order to determine the best parameters for this scenario, and apply them to get the best results.

It is important to mention that the focus is one the per device dataset, since there a full potential of federated learning can be seen. The other two subdatasets are more of a baseline and comparison approaches, since they would need centralized storage, and are more complicated approach due to the remote position of the livestock.

5.1 Full dataset

In this section, the results using the full dataset are shown. First, LSTM hyperparameters are shown to see which parameter combination is used for this subdataset. After, initial

metric used to show how well the models perform is distance error metric, followed with energy consumption and execution time for training and inference. Lastly, the visual representation of the results is shown and also the performance of different methods is intuitively shown to highlight which methods performed better than others.

5.1.1 Hyperparameter tuning

In Table 5.1, the hyperparameters that showed best performance on the dataset are shown.

Hyperparameter	Value
clipvalue	0.0
epochs	50
learning_rate	0.001
dropout_rate	0.0
units	64
layers	1

Table 5.1: Optimal Hyperparameters after Tuning TEMPLATE

Using these hyperparameters, the LSTM deep learning model will be trained to show the results in the following subsections.

5.1.2 Distance error metric

In Table 5.2, the distance Error metrics for all baseline and deep learning models is shown.

Model	Mean	SD	RMSE
ARIMA	183.6821	234.7211	298.0488
ARIMAX	**101.0661**	96.6107	**139.8141**
SARIMAX	253.7625	**89.1968**	268.9823
LSTM	120.7768	107.6015	161.7564

Table 5.2: Performance Metrics of Time Series Models for the Full dataset

Looking at all three metrics together, the best performing model in the case for the full dataset is ARIMAX. This is normally unusual, since the more advanced method of LSTM takes more time to train and is more complex normally produces much better results.

The only metric where ARIMAX is not performing the best is the Standard Deviation, and the only model that has a lower deviation is SARIMAX. However, the Mean and the RMSE are both much higher for SARIMAX, and thus not making it the best option for this dataset.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

5.1.3 Energy consumption and execution time for training and inference

In table 5.3, Energy consumption and execution time are shown for all models used to predict using the full dataset.

Model	Type	Energy (J) (CPU)	Execution time (s)
ARIMA	Training	26.6384	0.6321
ARIMA	Inference	0.2277	0.0062
ARIMAX	Training	**2.1569**	**0.0793**
ARIMAX	Inference	**0.1320**	**0.0074**
SARIMAX	Training	123.2973	2.6795
SARIMAX	Inference	0.8218	0.0196
LSTM	Training	780.9036	40.0756
LSTM	Inference	3.4700	0.2290

Table 5.3: Performance Metrics of Different Time Series Models

Again looking at all the metrics together, ARIMAX uses the least amount of energy and completes training the fastest. As for Inference, execution time is slightly slower from ARIMA, but insignificantly. Efficiency wise, it is the best method, even for both training and inference.

On the other hand, LSTM uses significantly more energy and execution time than any other model for the full dataset, making it not the ideal choice for the best model. Even though distance wise it was a close second to ARIMAX, the consumption doesn't justify the distance metrics.

5.1.4 Visualization of predictions

Since the results can be easily visualized on a map, it is a very good way to see how the results are different between two different methods.

In the case of the full dataset, the best two models are ARIMAX and LSTM, and the results can be seen in Figures 5.1 and 5.2. The red triangle represents the predicted points, and the green circles represent the actual points.

Looking at the actual points, there are three different clusters where the points are concentrated at. In comparison, the predicted points for LSTM are much more closer to the true values than the predicted values for ARIMAX in two clusters. Although, the third one overlaps much more for ARIMAX, and is probably the reason why the metrics show that ARIMAX is overall better.

The performance could be most likely explained that the model can't fully comprehend the natural movements of livestock when training with combined data, since the patters are combined and not easily extracted by the model.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

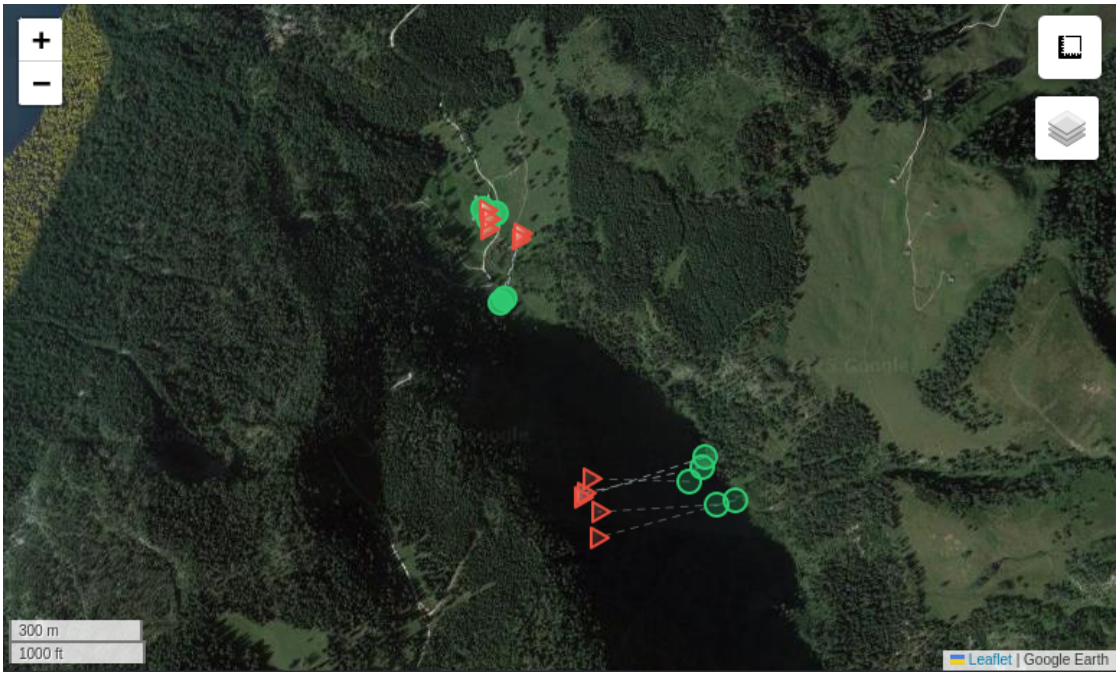

Figure 5.1: Predictions of ARIMAX model against the true values

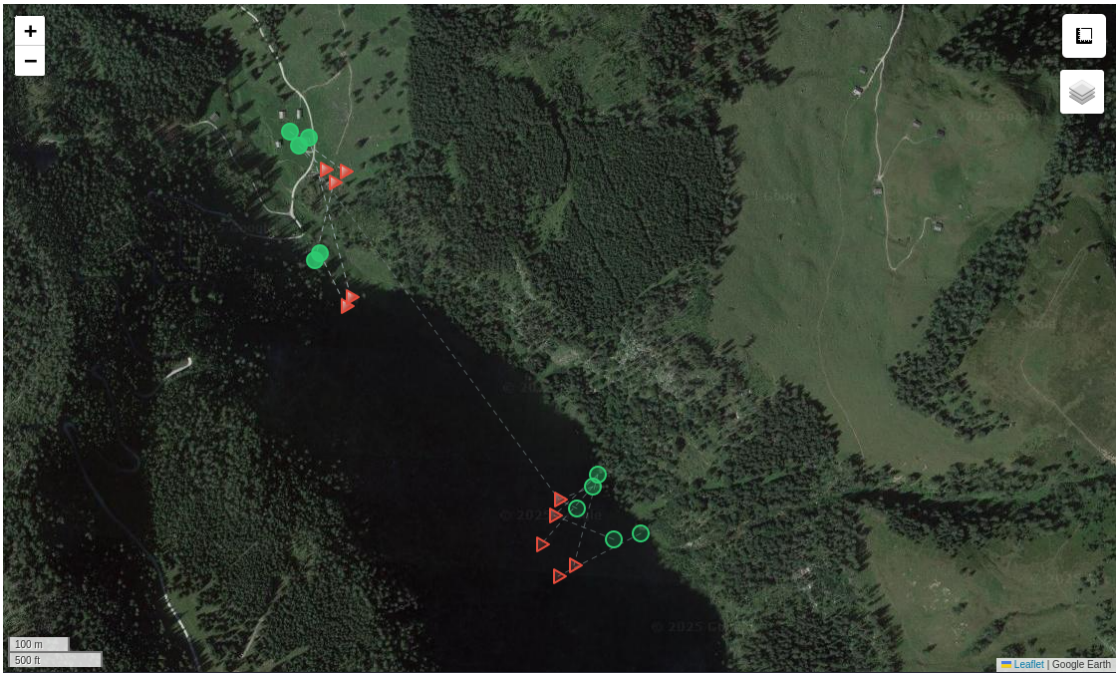

Figure 5.2: Predictions of LSTM model against the true values

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

5.2 Per device subdataset

5.2.1 Hyperparameter tuning

In Figure 5.3, the hyperparameters that showed best performance on the dataset are shown. These are the hyperparameters for LSTM, and for Federated learning version of the model is shown also in Figure 5.3.

Federated model has an additional parameters of rounds, which described as mentioned in Chapter 3, is a number of times weights are updated in the global model. Using these hyperparameters, the LSTM deep learning model and Federated model will be trained to show the results in the following subsections.

Hyperparameter	Value
clipvalue	0.0
epochs	50
learning_rate	0.001
dropout_rate	0.1
units	64

Hyperparameter	Value
clipvalue	0.0
epochs	1
learning_rate	0.001
dropout_rate	0.0
units	64
rounds	5

Figure 5.3: Optimal Hyperparameters for LSTM and Federated Learning LSTM Models.

5.2.2 Distance error metric

In Figure 5.4 are the results of all distance error metrics. In the first table are the results of three baseline methods, while in the second are the advanced approaches of LSTM and Federated Learning.

When comparing baseline methods with the previous approach, the results are slightly worse performing. On the opposite side, the advanced approaches of LSTM and Federated learning are performing much better than any other method up to now.

This could be due to the fact that the deep learning model is able to learn individual pattern movements from livestock, and convey them better in the predictions. This does not mean that the baseline approaches have bad results, only that the advanced approaches have better enough results to discuss the runtime trade-offs.

Next step is to look at energy consumption and execution time to determine the best model between the LSTM and Federated Learning one.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

ID	ARIMA			ARIMAX			SARIMAX		
	Mean	SD	RMSE	Mean	SD	RMSE	Mean	SD	RMSE
1	183.55	136.37	228.67	189.09	128.44	228.59	339.21	151.57	371.54
2	617.24	302.29	687.29	143.11	104.56	177.24	465.50	179.16	498.78
3	549.90	225.90	594.49	93.69	59.28	110.87	205.30	54.96	212.53
4	203.58	97.50	225.72	177.25	88.18	197.97	402.89	78.59	410.48
5	945.72	302.01	992.77	66.37	45.64	80.55	166.52	13.92	167.10
6	469.61	176.02	501.51	**62.80**	**44.57**	**77.00**	130.61	65.69	146.20
7	548.34	240.04	598.58	230.55	153.37	276.90	661.48	291.79	722.98
Avg	502.56	211.45	547.00	**137.55**	**89.15**	**164.16**	338.79	119.38	361.37

ID	LSTM			Federated		
	Mean	SD	RMSE	Mean	SD	RMSE
1	40.82	**33.39**	**52.74**	**39.62**	35.07	**52.91**
2	51.50	130.60	140.39	41.48	135.00	141.23
3	74.60	85.45	113.43	76.22	86.09	114.99
4	77.65	135.45	156.13	67.02	138.89	154.21
5	29.20	88.09	92.81	38.57	86.34	94.57
6	56.73	125.64	137.85	52.08	129.52	139.60
7	75.71	134.43	154.29	53.58	140.70	150.56
Avg	58.03	**104.72**	**121.09**	**52.65**	107.37	**121.15**

Figure 5.4: Performance Metrics of Time Series Models by Device ID.

5.2.3 Energy consumption and execution time for training and inference

In table 5.4, Energy consumption and execution time are shown for all models used to predict using the full dataset.

Comparing the two deep learning methods, even though they both have similar results, the federated learning approach has much faster both training and inference times while consuming a significantly smaller amount of energy. The additional advantage of federated learning over all other models for this subset is that only one model is necessary for making predictions. All other approaches need one model for each device, while a general model is built with federated learning. While for seven devices this is not a big problem, when scaling the number of devices makes using all different methods harder, the general model for federated leaning just becomes even better at predicting.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Model	Type	Energy (J)		Execution time (s)	
		Total	per-device	Total	per-device
ARIMA	Training	131.3544	18.7649	3.3842	0.4835
ARIMA	Inference	0.9826	0.1404	0.0291	0.0041
ARIMAX	Training	**19.6972**	**2.8139**	**0.8464**	**0.1209**
ARIMAX	Inference	**0.8679**	**0.1240**	**0.0403**	**0.0058**
SARIMAX	Training	1180.4381	168.6340	27.4270	3.9181
SARIMAX	Inference	4.9667	0.7095	0.1197	0.0171
LSTM	Training	2148.7064	306.9580	128.393	18.3418
LSTM	Inference	23.5800	3.3685	1.269	0.1812
Federated_LSTM	Training	**365.28**	**52.1828**	**11.16**	**1.59**
Federated_LSTM	Inference	**29.4279**	**4.2039**	**1.385**	**0.1978**

Table 5.4: Performance Metrics for per Device subset

5.2.4 Visualization of predictions

In order to see the another advantage of the federated approach, in Figures 5.5 and 5.6 is the comparison on the visualized results. The results are almost identical, once again showing not only is the performance not lost using the federated approach, but the method performs quite well.

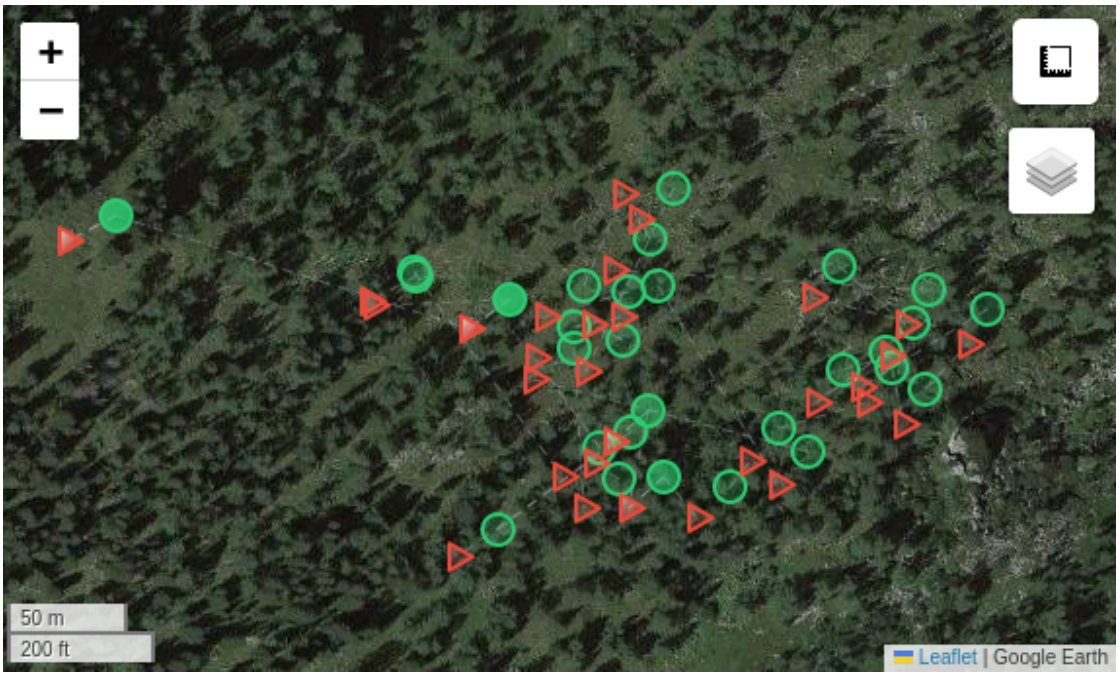

Figure 5.5: Predictions of LSTM model against the true values

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

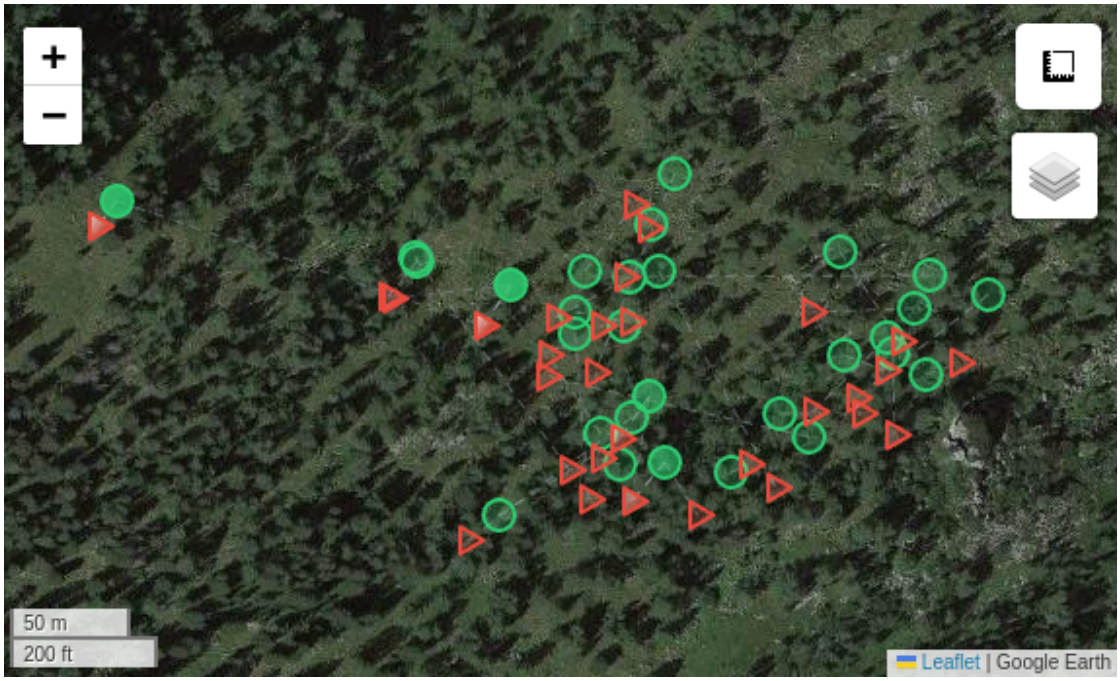

Figure 5.6: Predictions of Federated model against the true values

5.3 Day/Night subdataset

5.3.1 Hyperparameter tuning

In Table 5.1, the hyperparameters that showed best performance on the dataset are shown.

Hyperparameter	Value
clipvalue	0.0
epochs	50
learning_rate	0.001
dropout_rate	0.0
units	64

Table 5.5: Optimal Hyperparameters after Tuning TEMPLATE

Using these hyperparameters, the LSTM deep learning model will be trained to show the results in the following subsections.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

5.3.2 Distance error metric

In Figure 5.7 are the results of all distance error metrics. In the first table are the results of three baseline methods, while in the second is the advanced approach of LSTM.

Comparing the last data split to the previous ones, the results are worse over all three metrics over all methods. The best one out of them is again the baseline ARIMAX approach where it beats the LSTM methods over all three metrics.

Set	ARIMA		
	Mean	SD	RMSE
Day	764.19	268.55	810.00
Night	333.61	330.64	469.70
Avg	548.90	299.60	639.85

Set	ARIMAX		
	Mean	SD	RMSE
Day	272.59	95.34	288.78
Night	135.83	100.84	169.17
Avg	**204.21**	**98.09**	**228.98**

Set	SARIMAX		
	Mean	SD	RMSE
Day	766.17	199.83	791.80
Night	259.57	85.23	273.20
Avg	512.87	142.53	532.50

Set	LSTM		
	Mean	SD	RMSE
Day	267.88	140.94	302.70
Night	224.34	147.79	268.64
Average	246.11	144.37	285.67

Figure 5.7: Performance Metrics by Set (Day/Night) for ARIMA, ARIMAX, SARIMAX, and LSTM Models.

Comparing the two different parts of the dataset, the day and night, on average the night part has better performance, even though that the data comes in more regularly during the day while at night the data is received less frequently. This is probably due to the fact that naturally, livestock moves less at night, making it easier to make correct predictions.

The results still don't compare to the other two approaches, however, the energy analysis still remains.

5.3.3 Energy consumption and execution time for training and inference

Even though the performance is not up to par to the other splits of the data, the energy consumption and execution time for training and inference can still outperform them. In table 5.6, the energy consumption and execution time are shown for all models used to predict using the Day and Night data split.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Model	Type	Energy (J) (CPU)	per subset	Execution time (s)	per subset
ARIMA	Training	69.9471	34.9736	1.7482	0.8741
ARIMA	Inference	0.3876	0.1938	0.0101	0.0051
ARIMAX	Training	**6.4926**	**3.2463**	**0.2784**	**0.1392**
ARIMAX	Inference	**0.2233**	**0.1117**	**0.0113**	**0.0057**
SARIMAX	Training	251.9471	125.9736	5.9560	2.9780
SARIMAX	Inference	1.8420	0.9210	0.0447	0.0224
LSTM	Training	948.5259	474.2630	45.3156	22.6578
LSTM	Inference	8.6778	4.3389	0.4765	0.2383

Table 5.6: Performance Metrics per Sub-dataset

Compared to the alternative two methods, the metrics are better than the per device dataset, while it is slightly worse than the full dataset. Since the full dataset has better performance metrics, it is then a definitively better option since it beats the current approach on both ends.

Overall the approach has the worst metrics of all, and second best energy consumption and execution time, making it the least viable option to be used. However, it was a logical step to try due to the nature of the livestock movement.

5.4 Discussion and Limitations

To sum up, this chapter showed the results of the experiments of efficiency of location prediction in rural areas. Three different data split approaches were taken, where the focus is on the second one, since it has a real application in federated learning and edge computing.

The baselines, the full dataset and day/night split, showed two different results. The full dataset showed promising results, where the best model was ARIMA, even outperforming a deep learning approach LSTM. As for the day/night split, there were no methods that showed any promise, but it was a worthy try.

Lastly, the per device split gave the best results especially using the federated learning approach. It is the best method not only performance wise, but considering the convenience of the edge computing and general model aggregation.

A comprehensive analysis of the performance metrics across all models reveals substantial differences in efficiency and accuracy. The federated learning approach showed the lowest error metrics while having one of the best execution times and energy consumptions. While ARIMAX and LSTM had good performance federated learning's robustness makes it the most suitable solution for the target application.

Even though the experiments have been successfully ran and decent results have been achieved, there have been certain limitations that were impacting the work. The most

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

obvious and the most influential one is the data used. Since FFG Virtual Shepherd is still a project in development, there is not a whole lot of data, only for one season so far. Using multiple seasons would certainly give more information to the models and help them understand data better, and with that improve the performance even more.

Not only that, but since the devices used for location tracking are still in development, leading to often inaccurate locations. This can be mitigated through anomaly detection, but having more accurate positions will be beneficial since there won't be a need to locate anomalies and correct them. With time, this will certainly be improved which will in the end most likely lead to much better results.

Lastly, since anomaly detection was not the topic of the thesis, it was done in a very simple way just to mitigate the effect of very big anomalies that would badly skew the results. A more sophisticated method could immediately change the results for the better, since the data the models would be trained on would be much cleaner and would showcase true patterns that the livestock leaves in the movement.

Even though there certainly are limitations that have had a negative effect on the whole project, overall the experiments were successfully ran and the results are pretty good considering the difficulties.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 6

Conclusion and Future Work

To summarize and conclude, the thesis showed a combination of a deep neural network and federated learning to get the best results possible for location prediction.

The best results overall came from the per device subsdataset, especially from the deep learning methods, followed by the baseline ARIMAX. Even though Federated LSTM performs slightly better when considering only performance metrics of the models in comparison to only using LSTM, there are also even more important advantages to the method.

First, the execution time and energy consumption are much lower, while not only keeping the performance, but having and even better one as mentioned. This makes the utilization of federated learning highly effective, and very convenient as well. This is because there is only one global model when training in a federated manner, but when using only a LSTM deep learning model, one model per device is necessary in order to make all predictions. In case of many livestock devices, the models become hardly managed and not easy to maintain. In addition to having to keep track of all models, a central storage system is necessary to store all of the data and train all the models, while the federated approach only needs a central system to communicate between models during training, significantly cutting on communication costs. In this case the data is stored on devices locally, which in rural area is quite an advantage since it disrupts the need of sending the new position every 20 to 40 minutes.

Managing only one global model while cutting out a need for central storage and heavy communication, but still keeping decent energy consumption and execution time and having the best performance metrics is what federated learning helped achieve here in this project. Having many advantages while having virtually zero downsides is one more testament of why federated learning became such a popular method in such a short amount of time.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

This thesis also showed that it is not only effective in environments where privacy is the biggest concern, but also it is a great addition to edge computing environments in order to cut costs while keeping the necessary performance of the models.

Work so far is a very strong basis for some future work not only in this FFG Virtual Sherpherd project but in this field. There are things that could be done to directly expand this project but also do similar things in the spirit of the FFG project.

Starting with anomaly detection, a more advanced direction could be taken, for example using an Autoencoder on the data and to see if some values stand out. As previously mentioned, this would dramatically improve the results immediately for the thesis, and that is why it could be considered as a direct expansion to the current work.

In addition to this, an even more advanced neural network than LSTM could be used, in order to understand data patterns even better and thus improve the predictions. For example, a model with an attention mechanism could do this job very well, and thus would be a great choice for exactly that task.

A combination of these two improvements would surely give visible improvements to the current state of the project, and would surely be a great next step.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Overview of Generative AI Tools Used

In this work, AI tools were used for formatting tasks, such as LaTeX table modification and mathematical expression formation based on the inputs by the author. All conceptual development and writing was done by the author.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

List of Figures

2.1 Visualization of LSTM cell architecture 12

3.1 Visualization of the centralized architecture 21
3.2 Visualization of the decentralized architecture 21

4.1 Visualized all data points of all of the livestock 30
4.2 Visualized grazing area to represent the altitude differences 30
4.3 Visualized daily movement of one cow 32
4.4 Visualized data points of cows in their farm 33
4.5 Visualized *merge_asof* function 34
4.6 Haversine distance shown on the sphere 38

5.1 Predictions of ARIMAX model against the true values 46
5.2 Predictions of LSTM model against the true values 46
5.3 Optimal Hyperparameters for LSTM and Federated Learning LSTM Models. 47
5.4 Performance Metrics of Time Series Models by Device ID. 48
5.5 Predictions of LSTM model against the true values 49
5.6 Predictions of Federated model against the true values 50
5.7 Performance Metrics by Set (Day/Night) for ARIMA, ARIMAX, SARIMAX, and LSTM Models. 51

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

List of Tables

2.1 All Hyperparameters taken into consideration 14
2.2 Comparison of LSTM with ARIMA-family Models 15

4.1 Sensor Device Data 28
4.2 Selected Features for Position Prediction 28
4.3 Device Data 29
4.4 Number of data points in each subset 35
4.5 Number of points per device in the dataset 36
4.6 Coordinate Points 36
4.7 Difference Between Two Coordinate Points 37

5.1 Optimal Hyperparameters after Tuning TEMPLATE 44
5.2 Performance Metrics of Time Series Models for the Full dataset 44
5.3 Performance Metrics of Different Time Series Models 45
5.4 Performance Metrics for per Device subset 49
5.5 Optimal Hyperparameters after Tuning TEMPLATE 50
5.6 Performance Metrics per Sub-dataset 52

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

List of Algorithms

1
Averaging (FedAvg)24

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Bibliography

[Aka74] Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

[BD16] P. J. Brockwell and R. A. Davis. Introduction to time series and forecasting. 2016.

[BEG+19] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, H Brendan McMahan, et al. Practical secure aggregation for privacy-preserving machine learning. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2019.

[BIK+17a] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 1175–1191, New York, NY, USA, 2017. Association for Computing Machinery.

[BIK+17b] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[BJ70] George EP Box and Gwilym M Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day, 1970.

[BJRL15] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. Time series analysis: Forecasting and control. 2015.

[CH95] Fabio Canova and Bruce E Hansen. Are seasonal patterns constant over time? a test for seasonal stability. *Journal of Business & Economic Statistics*, 13(3):237–252, 1995.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

[CLD+18] Fei Chen, Mi Luo, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. Federated meta-learning with fast convergence and efficient communication. *arXiv preprint arXiv:1802.07876*, 2018.

[CNS23] Marina Costantini, Giovanni Neglia, and Thrasyvoulos Spyropoulos. Feddec: Peer-to-peer aided federated learning, 2023.

[Cor23] Intel Corporation. Running average power limit (rapl) energy reporting advisory guidance. `https://www.intel.com/content/www/us/en/developer/articles/technical/software-security-guidance/advisory-guidance/running-average-power-limit-energy-reporting.html`, 2023.

[CW15] Noel Cressie and Christopher K. Wikle. Statistics for spatio-temporal data. *Wiley*, 2015.

[DF79] David A Dickey and Wayne A Fuller. Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American Statistical Association*, 74(366a):427–431, 1979.

[FFG24] FFG Projektdatenbank. Virtual shepherd - satellitengestütztes monitoring von viehherden im alpinen raum. `https://projekte.ffg.at/projekt/5122077`, Februar 2024.

[FK18] Thomas Fischer and Christopher Krauss. Applications of lstm networks for time series forecasting. *The Journal of Financial Data Science*, 1(1):1–13, 2018.

[GKN17] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[GSK+17] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2017.

[HA21] R. J. Hyndman and G. Athanasopoulos. Forecasting: Principles and practice. 2021.

[HK08] Rob J Hyndman and Yeasmin Khandakar. Automatic time series forecasting: the forecast package for r. *Journal of Statistical Software*, 27(3):1–22, 2008.

[HRM+18] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

[HZL+19] Yuxiu Hua, Zhifeng Zhao, Rongpeng Li, Xianfu Chen, Zhiming Liu, and Honggang Zhang. Deep learning with long short-term memory for time series prediction. *IEEE Communications Magazine*, 57(6):114–119, 2019.

[KMY+16] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[KPSS92] Denis Kwiatkowski, Peter CB Phillips, Peter Schmidt, and Yongcheol Shin. Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of Econometrics*, 54(1-3):159–178, 1992.

[KYT+20] Latif U Khan, Ibrar Yaqoob, Nguyen H Tran, Syed Muhammad Ahsan Kazmi, Trung N Dang, and Choong Seon Hong. Edge-computing-enabled smart cities: A comprehensive survey. *IEEE Internet of Things Journal*, 7(10):10200–10232, 2020.

[LKX+20] Yang Liu, Yan Kang, Chaoping Xing, Tianjian Chen, and Qiang Yang. Federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[LKZ+18] Yuxuan Liang, Songyu Ke, Junbo Zhang, Xiuwen Yi, and Yu Zheng. Geoman: Multi-level attention networks for geo-sensory time series prediction. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI-18)*, pages 3428–3434, 2018.

[LPM15] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[LSZ+20] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks, 2020.

[LWL+16] Wei Liu, Zhiyong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. Deep learning for spatio-temporal data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 28(10):2259–2273, 2016.

[LWWT16] Qiang Liu, Shuang Wu, Liang Wang, and Tieniu Tan. Predicting the next location: A recurrent model with spatial and temporal contexts. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1):194–200, 2016.

[LYLL21] Yang Liu, Han Yu, Jian Liu, and Tianyi Liu. When federated learning meets quantum machine learning. *IEEE Transactions on Computational Social Systems*, 2021.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

[MMR+17] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. *Artificial intelligence and statistics*, pages 1273–1282, 2017.

[Pan91] Alan Pankratz. *Forecasting with dynamic regression models*. John Wiley & Sons, 1991.

[pdt24] The pandas development team. pandas.DataFrame.merge_asof. `https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.merge_asof.html`, 2024. Accessed 2025.

[RHL+20] Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletarì, Holger R Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N Galtier, Bennett A Landman, Klaus Maier-Hein, et al. The future of digital health with federated learning. *NPJ digital medicine*, 3(1):1–7, 2020.

[S+78] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[SCW+15] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015.

[SCZ+16a] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

[SCZ+16b] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

[SLG+22] Shanshan Song, Jun Liu, Jiani Guo, Chuang Zhang, Tingting Yang, and Junhong Cui. Efficient velocity estimation and location prediction in underwater acoustic sensor networks. *IEEE Internet of Things Journal*, 9(4):2984–2998, 2022.

[SP97] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[The24] The pandas development team. pandas-dev/pandas: Pandas. `https://github.com/pandas-dev/pandas`, April 2024.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus,

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[WFW+17] Fan Wu, Kun Fu, Yang Wang, Zhibin Xiao, and Xingyu Fu. A spatial-temporal-semantic neural network algorithm for location prediction on moving objects. *ISPRS International Journal of Geo-Information*, 6(4):99, 2017.

[WJH22] Dingzhu Wen, Ki-Jun Jeon, and Kaibin Huang. Federated dropout—a simple approach for enabling federated learning on resource constrained devices. *IEEE Wireless Communications Letters*, 11(5):923–927, 2022.

[WWPP19] Zhiwei Wu, Chong Wang, Wen Peng, and Shaofu Pan. A hybrid model for short-term residential load forecasting based on lstm and attention mechanism. *IEEE Access*, 7:112481–112490, 2019.

[WZC+18] Dong Wang, Junbo Zhang, Wei Cao, Jian Li, and Yu Zheng. Urban mobility prediction: A deep learning approach. *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI-18)*, pages 2576–2582, 2018.

[XKG19] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Asynchronous federated learning. *arXiv preprint arXiv:1903.03934*, 2019.

[XYT+21] Qi Xia, Winson Ye, Zeyi Tao, Jindi Wu, and Qun Li. A survey of federated learning for edge computing: Research problems and solutions. *High-Confidence Computing*, 1(1):100008, 2021.

[YLCT19a] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 36(3):50–60, 2019.

[YLCT19b] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[YTW+19] Huaxiu Yao, Xianfeng Tang, Hua Wei, Yu Zheng, and Zhenhui Li. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(1):5668–5675, 2019.

[ZCL+19] Zhi Zhou, Xu Chen, En Li, Liekang Zeng, Ke Luo, and Junshan Zhang. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, 107(8):1738–1762, 2019.

[Zhe15a] Yu Zheng. Trajectory data mining: An overview. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3):1–41, 2015.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

[Zhe15b] Yu Zheng. Urban computing: Concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3):1–55, 2015.

[ZZL+21] Weishan Zhang, Tianwei Zhou, Qinghua Lu, Xin Wang, Chunsheng Zhu, Hailiang Sun, Zhi Wang, Sin Kit Lo, and Fei Chen. Federated learning for smart manufacturing: A federated learning framework for cyber-physical-system enabled smart manufacturing. *IEEE Internet of Things Journal*, 2021.

[ZZL+24] Yifei Zhang, Dun Zeng, Jinglong Luo, Xinyu Fu, Guanzhong Chen, Zenglin Xu, and Irwin King. A survey of trustworthy federated learning: Issues, solutions, and challenges. *ACM Trans. Intell. Syst. Technol.*, 15(6), October 2024.

[ZZQ17] Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1):1655–1661, 2017.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.