Bachelor's thesis

Machine Learning-based User Movement Prediction in Layer 2 Networks

Vorhersage von Benutzerbewegungen in Layer 2 Netzwerken basierend auf Maschinellem Lernen

by

Lina Wilske

Supervisors

Prof. Dr. Holger Karl

Leonard Paeleke

Internet Technology and Softwarization Group

Hasso Plattner Institute at University of Potsdam

August 17, 2023

Abstract

...

Contents

1 Introduction **1**

2 Background **3**
- 2.1 Machine Learning in Wireless Networks 3
- 2.2 Time Series Prediction 3
- 2.3 Supervised Learning 4
- 2.4 Classification 4
- 2.5 Univariate and Multivariate Time Series 4
- 2.6 Temporal Dependency Handling 4

3 Dataset analysis and preparation **5**
- 3.1 Components of the dataset 5
- 3.2 File structure 5
- 3.3 Improvement on data for an ML model 7
- 3.4 Peculiarities of the data 7

4 Suitable Machine Learning Algorithm **11**
- 4.1 Classification Models 11
  - 4.1.1 Multilayer Perceptron (MLP) 11
  - 4.1.2 Hidden Markov Model (HMM) 11
  - 4.1.3 Recurrent Neural Network (RNN) 12
    - 4.1.3.1 Long Short-Term Memory (LSTM) 12
  - 4.1.4 Discussion of Classification Models 12

5 Implementation **15**
- 5.1 Preprocessing 15
- 5.2 Long Short-Term Memory (LSTM) 15
- 5.3 Training 15

6 Evaluation **19**
- 6.1 Adapting parameters 19

7 Conclusion **21**
- 7.1 Conclusion 21

References **23**

1 Introduction

In large-scale Wireless Fidelity (Wi-Fi) environments such as office buildings, shopping malls, and airports, where multiple Access Points (APs) are required, people often move around indoors with their mobile devices. To maintain a stable connection to the Service Set Identifier (SSID) a device connects to, it must remain in the range of the AP or may roam to another AP with the same SSID. Roaming has been an essential feature of Wi-Fi since the advent of the 802.11k[5] feature. This process improved with AP-initiated roaming, introduced in 802.11r[4]. However, the current roaming process does not account for human movement patterns. For example, if a station is moving away from AP1 towards AP2, ideally, AP2 should initiate the roaming process. Furthermore, an AP may instruct a client device to roam based on signal strength without considering the device's trajectory or the user's likely destination. The existing solutions need to be more robust and often lead to frequent hand-offs, inefficient resource usage, and in some cases, dropped connections. This thesis will explore using time series Machine Learning (ML) model to predict the top 3 Basic Service Set Identifiers (BSSIDs) a station may connect to next. Therefore, this thesis needs data with Wi-Fi, waypoint of clients and sensor data, e.g. acceleration.

A time series ML models require time series data as input. There are two possible data sources: generate new or utilize existing data. Data generation needs a comprehensive plan for accounting data setup and collection. The generation is a time-consuming process and needs a lot of planning and evaluation beforehand. Therefore, this thesis will utilize pre-existing data from a 2021 competition by Microsoft Research[6]. The data will be analyzed to determine what specific parts of the data we will use for the ML model.

Furthermore, the data will be prepared for a time series ML model. After that, we will discuss the suitability of some pre-selected ML models for the task. Due to the large number of data, this thesis will implement improving it, and evaluate the model's performance for a specific site and floor.

2 Background

2.1 Machine Learning in Wireless Networks

As Szott et al. noted in their survey[14], ML is also now utilized in Wireless Local Area Networks (WLANs). For Wi-Fi, a prediction for a possible roam could initiate the roaming process sooner, thereby improving the user experience and overall Wi-Fi network performance.

2.2 Time Series Prediction

Time series data, comprising a sequence of data points ordered in time, represents a common structure in many domains, including user mobility within a Wi-Fi network. Owing to its inherent temporal dependencies—where subsequent data points can be influenced by previous ones—particular machine learning techniques are typically employed. These include the Autoregressive Integrated Moving Average (ARIMA) model, and Recurrent Neural Network (RNN) models such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). Each of these techniques is designed to capture and leverage temporal patterns within the data, facilitating the prediction of future trends based on historical observations.

In the context of machine learning model development, the configuration of hyperparameters represents a crucial task. Defined as the set of parameters that govern the learning process and are not learned from the data, hyperparameters encompass elements such as the learning rate, the number of layers within a neural network, and the number of clusters in a clustering algorithm. As these parameters are determined a priori, their careful selection—known as hyperparameter tuning or optimization—is necessary to maximize model performance. This iterative procedure involves exploring various hyperparameter combinations in search of the configuration that yields the most accurate predictions.

Moreover, the evaluation of data stands as an essential component of any machine learning project, and this remains true for time series prediction. Such evaluation involves an assessment of the model's performance by contrasting predicted outcomes with actual results. One common technique is cross-validation, wherein the data set is partitioned into a training set for model training, and a validation set for model evaluation. Performance evaluation is indispensable for understanding the model's accuracy, reliability, and ability to generalize to new, unseen data. Furthermore, it provides insights into potential underfitting, where the model fails to learn sufficient patterns from the training data, or overfitting, where the model becomes overly sensitive to noise or outliers in the training data, both of which can significantly impair predictive performance.

2.3 Supervised Learning

Supervised learning is a ML technique that uses labeled data to train a model. The model learns from the training data and applies this learning to new, unseen data which is called test data.

2.4 Classification

A classification model in machine learning is a type of predictive model that categorizes incoming input data into specific categories or classes. It works by learning from a set of input features and corresponding labels during the training phase. It then applies this learning to new, unseen data. Classification models are used across various domains. The output of a classification model is discrete, meaning it assigns each input to a specific category. Examples of classification models include logistic regression, decision trees, random forests, and support vector machines.

2.5 Univariate and Multivariate Time Series

- Differ in number of variables
- univariate: one observation recorded sequentially over time, e.g., temperature, stock prices; focus on understanding and forecasting a single variable's behavior
- multivariate: multiple observations recorded over the same time intervals, allowing for the analysis of interrelationships and interdependencies between these variables, e.g. temperature and humidity; focus on delving into understanding dynamic interactions and co-movements between multiple variables

2.6 Temporal Dependency Handling

Temporal dependency handling refers to the ability of a model to recognize and leverage the relationships or dependencies between data points that are separated by time. In time series data, the value at a given time point can be influenced by previous values, and understanding this dependency is crucial for accurate predictions.

3 Dataset analysis and preparation

The dataset used in this thesis is the Indoor Location & Navigation from kaggle[10] which was part of a competition of Microsoft Research in 2021[6]. The data was recorded in shopping malls by the company XYZ[10] and was provided by Microsoft Research for this competition. The goal for the competition was, given a site-path file, predict the floor and waypoint locations at a timestamp given in the submission files. In the following, the dataset and data will be analyzed.

3.1 Components of the dataset

As noted in the kaggle notebook "Indoor Navigation: Complete Data Understanding" [8] the data consists of 3 parts:

- a train folder with train path files, organized by site and floor
- a test folder with test path files, organized by site and floor but without waypoint data
- a metadata folder with floor metadata, organized by site and floor, which includes floor images, further information and a geojson map

The train folder contains 204 subfolders, which represent each site where the data was recorded. In each site folder are a minimum of one and a maximum of twelve subfolders, which represent the floors of the site, the median is 5 floors. Overall there are 26,925 files each containing the movement of one person for a specific site and floor. Per floor there are between one and 284 files with a median of 14. The floor F1 of site 银泰城(城西店) which was hashed as "5d27075f03f801723c2e360f" in the train folder of the competition, has the most files.

For this thesis, the submission files as well as the test folder will not be used, because our goal is not to predict the floor and site name for a certain timestamp, but to predict the BSSID to which a device may connect next. Therefore, we will not analyze the content of these folders in more detail.

3.2 File structure

Each file in each floor folder is a **.txt** file. The first two lines and the last are denoted with "#". The first contains the start time of the recording, the second site information SiteID as hash, SiteName, FloorId as hash and FloorName. The last line contains the end time of the recording. The main part of the data consists of the collected data. Each line contains a UNIX timestamp in milliseconds, followed by a data type and the data itself, which are all separated by a tabulator. The GitHub repository of the competition[7] shows that the data type in the second column followed by its data can be one of the following:

(1) TYPE_ACCELEROMETER with x, y and z acceleration and an accuracy value
(2) TYPE_MAGNETIC_FIELD with x, y and z magnetic field and an accuracy value
(3) TYPE_GYROSCOPE with x, y and z gyroscope and an accuracy value
(4) TYPE_ROTATION_VECTOR with x, y and z rotation vector and an accuracy value
(5) TYPE_MAGNETIC_FIELD_UNCALIBRATED with x, y and z magnetic field and an accuracy value
(6) TYPE_GYROSCOPE_UNCALIBRATED with x, y and z gyroscope and an accuracy value
(7) TYPE_ACCELEROMETER_UNCALIBRATED with x, y and z acceleration and an accuracy value
(8) TYPE_WIFI with SSID, BSSID, Received Signal Strength Indication (RSSI), frequency, and last seen timestamp of the access point. The SSID and BSSID are hashed.
(9) TYPE_BEACON with Universally Unique Identifier (UUID), Major Identifier (MajorID), Minor Identifier (MinorID), Transmission Power (TxPower), RSSI, distance to the device measured by the beacon, Media Access Control (MAC) address and a timestamp as padding data. The MajorID and MinorID are hashed.
(10) TYPE_WAYPOINT with x and y coordinates which are the ground truth location labeled by the surveyor

```
#   startTime:1571462193934
#   SiteID:5d27099303f801723c32364d SiteName:银泰百货(庆春
     店) FloorId:5d27099303f801723c323650  FloorName:4F
1571462193944   TYPE_WAYPOINT   57.885998   69.501526
1571462194071   TYPE_ACCELEROMETER  -0.95254517 0.7944031   8.928757    2
1571462194071   TYPE_MAGNETIC_FIELD -25.65918   -4.4784546  -28.201294  3
1571462194071   TYPE_GYROSCOPE  -0.22373962 -0.07733154 -0.16847229 3
1571462194071   TYPE_ROTATION_VECTOR    0.04186145  -0.02101801 -0.72491926 3
1571462194071   TYPE_MAGNETIC_FIELD_UNCALIBRATED    -4.8568726  10.406494   -387.44965  20.802307
     14.884949   -359.24835  3
1571462194071   TYPE_GYROSCOPE_UNCALIBRATED -0.22218323 -0.068359375    -0.1628418  0.0026245117
     9.765625E-4 -7.6293945E-4   3
1571462194071   TYPE_ACCELEROMETER_UNCALIBRATED -0.95254517 0.7944031   8.928757    0.0 0.0 0.0 3
...
1571462194883   TYPE_WIFI   b06c4e327882fab58dfa93ea85ca373a54e887b5    9
     f967858afcbb907af6e5adef766c7e7b936ef07    -63 2462    1571462190744
1571462194883   TYPE_WIFI   8204870beb9d02995dab3f08aad97af5eab723cc    0413
     b35df78fc865af15b4721d5aeb33ff57da45    -64 2447    1571462188686
...
1571462194020   TYPE_BEACON 07efd69e3167537492f0ead89fb2779633b04949
     b6589fc6ab0dc82cf12099d1c2d40ab994e8410c    76e907e391ad1856762f70538b0fd13111ba68cd    -57 -71
      5.002991815535578   1b7e1594febd760b00f1a7984e470867616cee4e    1571462194020
...
#   endTime:1571462195976
```

Listing 3.1: A snippet from the dataset of a file of the floor F4 of the site with the ID 5d27099303f801723c32364d

Each file contains a different amount of waypoints and sensor data. The first and last data type in each file is a (10). Lines with types from (1) to (7) occur every 20 ms and are measured at the same time. (8) occurs about every 1800-2200 ms. (10) data is not evenly distributed. An assumption for this is, that the recording of the waypoint data is triggered by an exterior event, e.g. a button press. As seen in listing 3.1, the data are measured separately from each other, so there are no combinations of the data types.

A prediction of the next BSSID will only work per site, due to the different architectures of the sites. Still, prediction could be difficult for a whole site, because the APs are different on each floor which may result in many APs for the prediction. To get better results in the prediction, we will focus on a single floor of a site. The table 3.1 shows an analysis of the site with the most files for a single floor.

Information	**Value**
Total data points	7,157,081
Average data points per file	25,201
Number of waypoints	2,027
Lines of each (1) to (7) data	746,689
Lines of Wi-Fi data	1,862,044
Lines of beacon data	66,187
Number of BSSIDs	4,795
Number of APs	4,795
Number of SSIDs	1,421
RSSI range	-93 to -13 dBm

Table 3.1: Summary of data for F1 of site 银泰城(城西店)

3.3 Improvement on data for an ML model

As seen in previous sections, a location for the time of *TYPE_WIFI* data points is not provided. Between *TYPE_WAYPOINT* and *TYPE_WIFI* data further human movement may have occurred. However, they can be combined using linear interpolation, as seen in fig. 3.2.

Thus, *TYPE_WAYPOINT* and *TYPE_WIFI* are combined to get a location for the Wi-Fi data point. The interpolation results in 6549 waypoints which is 3 times more than the original amount of waypoints, as seen in fig. 3.3. Furthermore, we will also interpolate *TYPE_ACCELEROMETER* to improve the predictions. Now a multivariate time series is interpolated out of the original data, which will be used for the machine learning model. A further interpolation is possible, but we will focus on a simpler multivariate time series.

3.4 Peculiarities of the data

Further analysis of the dataset revealed some peculiarities, which are described in the following.

The data is collected by different devices, at different timestamps and days. A problem for the ML could be, that the waypoint data were measured irregularly.

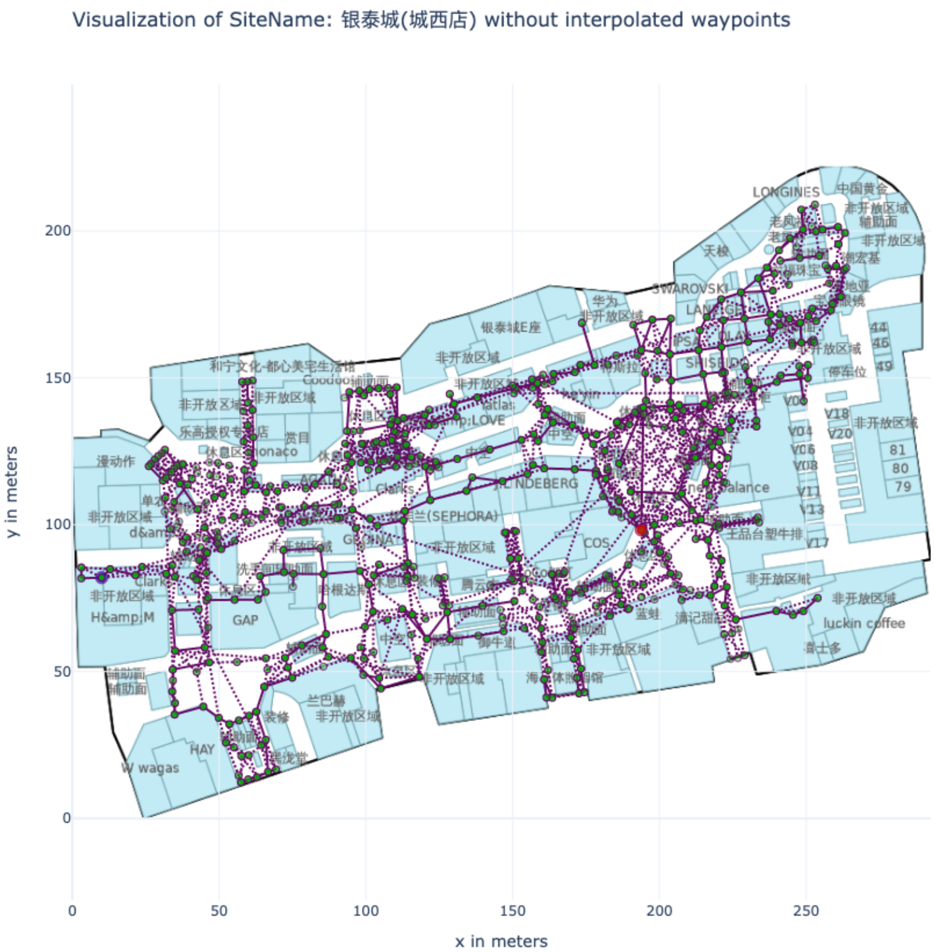


Figure 3.1: Visualization of the waypoints for SiteName: 银泰城(城西店)

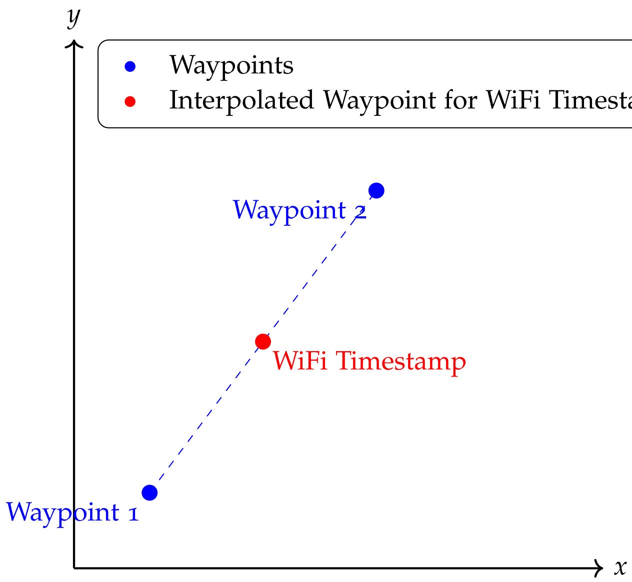


Figure 3.2: Visualization of linear interpolation for Wi-Fi timestamps based on given waypoints. The blue points represent the original waypoints, while the red points show the interpolated positions for specific Wi-Fi

- As in fig. 3.1 some waypoints seem to be very distant from the next one
- listing 3.2 shows the top 10 pairs of waypoints with the most significant metric differences
- if points are more than 10 meters apart, we will define them as "to apart from each other"
- split up the data where the distance between two points is more than 10 meters
- result: 123 files, with interpolation

```
1. Point 1: (247.96523998265695, 168.7631635050295), Point 2: (117.92375106521739,
     51.997759545341616), Metric Difference: 174.77113148839442
2. Point 1: (98.66346, 127.5971), Point 2: (258.75049789436116, 181.23350740899357), Metric
     Difference: 168.83342057049657
3. Point 1: (189.58672, 71.454666), Point 2: (89.73448203762376, 102.255128190099), Metric
     Difference: 104.49467879858156
4. Point 1: (223.49295, 145.0939), Point 2: (174.26284532732006, 78.86335505811792), Metric
     Difference: 82.52325908119289
5. Point 1: (34.864815, 35.45561), Point 2: (33.284438514193546, 110.76117936967742), Metric
     Difference: 75.32215057954856
6. Point 1: (50.31085719185683, 92.03105531572366), Point 2: (114.97229034709193,
     123.04521228267667), Metric Difference: 71.71456525741291
7. Point 1: (150.91972285390713, 145.15169976783693), Point 2: (222.23440919809525,
     146.11043967333333), Metric Difference: 71.32113060360388
8. Point 1: (64.64140228107132, 25.345204272946134), Point 2: (34.47475562023909, 84.44615420072283)
     , Metric Difference: 66.35471990088624
9. Point 1: (56.83799274253731, 74.52090035349569), Point 2: (94.43750948519768, 125.97292215289488)
     , Metric Difference: 63.72624425248555
10. Point 1: (172.3439286324042, 56.200716101045295), Point 2: (212.4478039192399,
     99.33967479470648), Metric Difference: 58.90068395354572
```

Listing 3.2: Top 10 pairs with the most significant metric differences

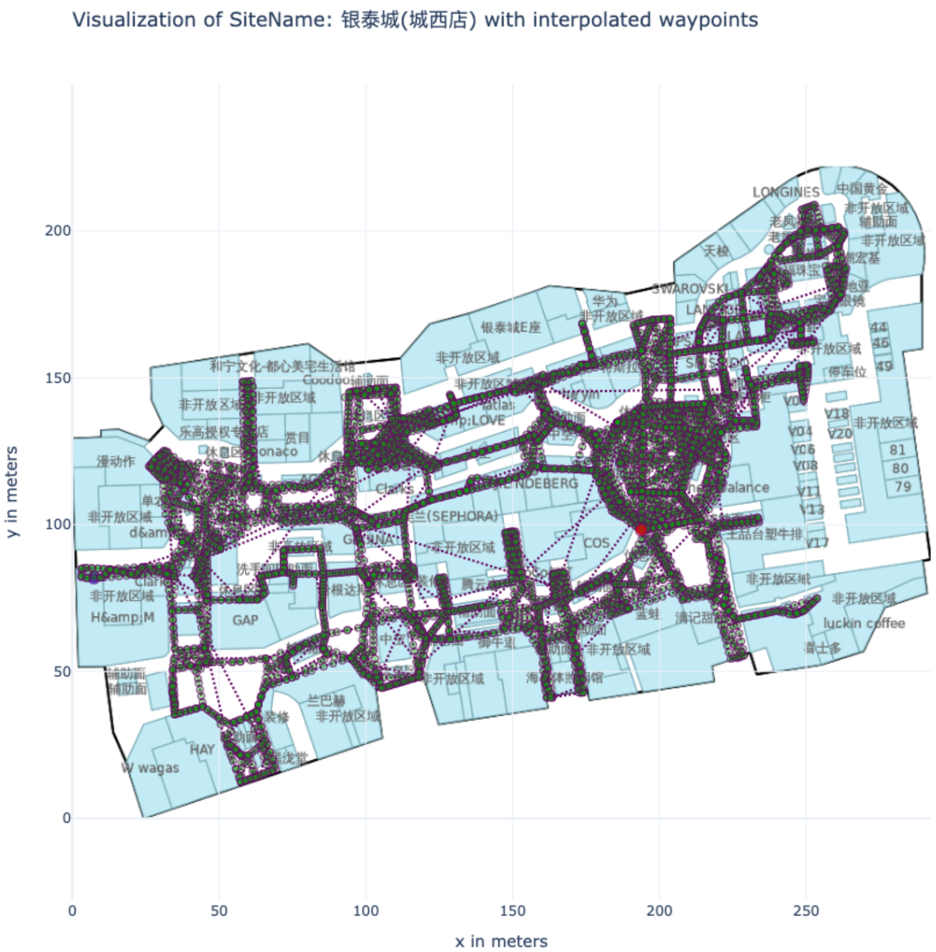


Figure 3.3: Visualization of the interpolated waypoints for SiteName: 银泰城(城西店)

4 Suitable Machine Learning Algorithm

- As seen in the previous chapter, a lot of data for one floor
- As this is a time series, we want to use a time series machine learning model
- As we want to predict the next BSSID out of many BSSIDs, so classification problem with supervised learning
- Discussion of some pre-chosen models and decision for one which will be implemented in the next chapter

4.1 Classification Models

- Stated in chapter 2, classification model in ML is a type of predictive model that categorizes incoming input data into specific classes
- Prediction of next BSSID is a classification problem
- therefore, interpret problem as classification problem
- For multivariate time series classification models there are only a few models
  - Multilayer Perceptrons (MLPs) [9]
  - RNNs such as LSTMs[3]
  - Hidden Markov Models (HMMs) [12],

4.1.1 MLP

MLP, also known as a feedforward artificial neural network, is a class of deep learning models, primarily used for supervised learning tasks. An MLP consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. Each node in one layer is connected with certain weights to every node in the following layer. MLPs apply a series of transformations, typically nonlinear, to the input data using activation functions, such as the sigmoid or Rectified Linear Unit (ReLU), facilitating the model's ability to model complex patterns and dependencies in the data [2].

4.1.2 HMM

HMM is a statistical model that assumes the system being modeled is a Markov process with unobserved (hidden) states[13]. HMMs are particularly known for their application in temporal pattern recognition such as speech, and handwriting. They describe the probability of a sequence of observable data, which is assumed to be the result of a sequence of hidden states, each producing an observable output according to a certain probability distribution.

4.1.3 RNN

RNN is a type of artificial neural network well-suited to sequential data because of its intrinsic design. Unlike traditional feedforward neural networks, an RNN possesses loops in its topology, allowing information to persist over time. This unique characteristic enables the model to use its internal state (memory) to process sequences of inputs, making it ideally suited for tasks involving sequential data such as speech recognition, language modeling, and time series prediction[1].

4.1.3.1 LSTM

LSTM is a special kind of RNN, capable of learning long-term dependencies, which was introduced by Hochreiter and Schmidhuber in 1997[3]. LSTMs were designed to combat the "vanishing gradient" problem in traditional RNNs, which made it difficult for these networks to learn from data where relevant events occurred with large gaps between them. The key to the LSTMs ability is its cell state and the accompanying gates (input, forget, and output gate), which regulate the flow of information in the network.

4.1.4 Discussion of Classification Models

As mentioned in chapter 3, the floor analyzed there has 4795 BSSIDs. So we have 4795 classes for the classification problem. The selection of the right model for this task is even more important.

We will discuss the classification models by the following topics: Temporal Dependency Handling, Capacity and Complexity, Multivariate Data, Flexibility and Integration, Regularization and Overfitting, and Adaptability.

Temporal Dependency Handling:

- LSTMs, by design, are equipped to handle long-term temporal dependencies. Their unique cell state and gating mechanisms allow them to store, modify, and access information over extended periods, making them adept at capturing patterns from long sequences.
- MLPs lack a built-in mechanism for remembering past information, making them less suitable for time series data where temporal order and dependencies are crucial.
- While HMMs can handle temporal dependencies to some extent, they often struggle with longer sequences and multivariate data due to their Markovian assumption, which limits their memory to the most recent state.
- Standard RNNs were designed to handle temporal dependencies, but they suffer the vanishing gradient problem, making them less effective in capturing long-term dependencies compared to LSTMs[11].

Capacity and Complexity: With 4,795 classes, the model needs a considerable capacity to differentiate between the subtle differences in patterns that might exist among them. LSTMs, being deep learning models, can scale effectively in terms of capacity by adding more layers or units, while still maintaining their ability to handle temporal data.

Multivariate Data:

- LSTMs can seamlessly handle multivariate time series data. Their recurrent nature allows them to process each time step with multiple features effectively.
- While MLPs can also handle multivariate data, they treat each feature and time step independently, often missing out on the interdependencies.
- HMMs are primarily designed for univariate data. Extending them to multivariate scenarios requires additional complexities and assumptions.

Flexibility and Integration: LSTMs can be easily integrated with other deep learning architectures, such as Convolutional Neural Networks (CNNs), to capture both temporal and spatial features. This flexibility is particularly useful when dealing with complex and varied data sources.

Regularization and Overfitting: Deep learning models, including LSTMs, come with a plethora of regularization techniques, such as dropout, which can be crucial when dealing with many classes and the risk of overfitting.

Adaptability: LSTMs can be trained with various architectures, like bidirectional LSTMs, which process the data from both past-to-future and future-to-past directions, providing a richer representation of the data.

In conclusion, while MLPs, HMMs, and traditional RNNs have their strengths and have been successful in many applications, they have problems with multivariate time series classification with a vast number of classes. This problem demands a model that can capture intricate temporal patterns, scale in capacity, and handle multivariate data efficiently. LSTMs, with their unique architecture and properties, can deal with this challenge, making them the preferred choice for this task and the selected model for our implementation.

5 Implementation

5.1 Preprocessing

- data from chapter 3 is used for preprocessing
- Load data from the CSV files.
- Concatenate data from the first 10 lines of each file.
- Replace missing BSSID value
- Create a target variable.
- Normalize the data.
- Create sequences of data based on $window_size.Encodethetargetvariable$.
- Split the data into training and testing sets.

5.2 LSTM

- Decision in previous chapter for LSTM
- LSTM layer with 1000 units.
- Dense layer with softmax activation (number of units equal to the number of categories in the target variable).

5.3 Training

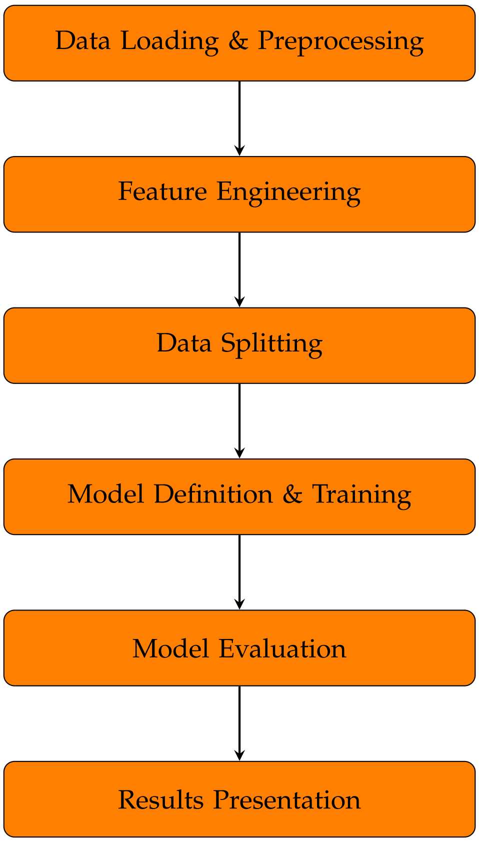
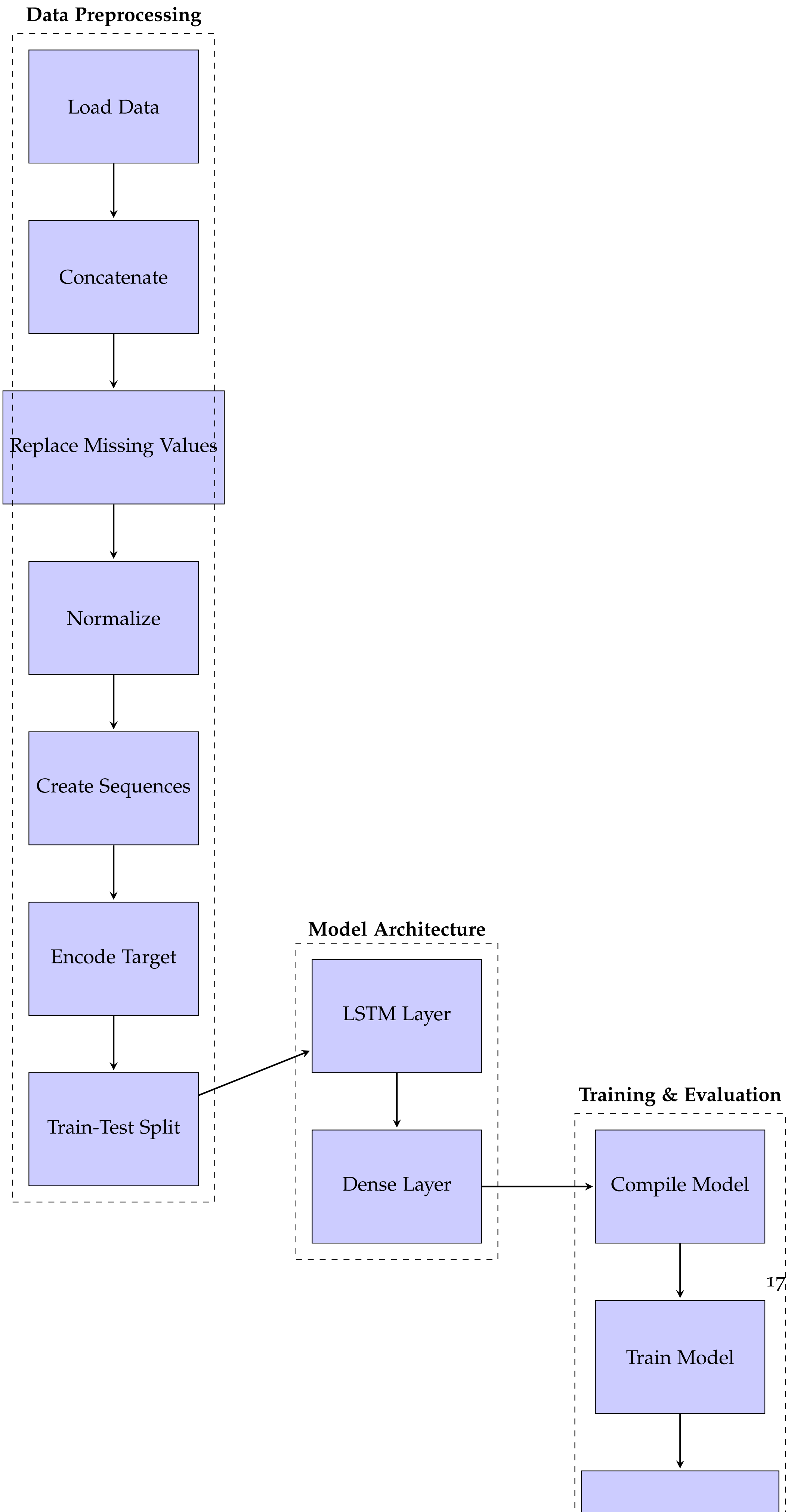


Figure 5.1: Flow diagram of the LSTM implementation process.


Data Preprocessing
Load Data
Concatenate
Replace Missing Values
Normalize
Create Sequences
Encode Target
Train-Test Split
Model Architecture
LSTM Layer
Dense Layer
Training & Evaluation
Compile Model
Train Model

6 Evaluation

6.1 Adapting parameters

- ...

7 Conclusion

7.1 Conclusion

- ...

Bibliography

[1] Jeffrey L. Elman. "Finding Structure in Time". en. In: *Cognitive Science* 14.2 (Mar. 1990), pages 179–211. ISSN: 03640213. DOI: 10.1207/s15516709cog1402_1. URL: http://doi.wiley.com/10.1207/s15516709cog1402_1 (visited on Aug. 6, 2023).

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. eng. OCLC: 987005922. Cambridge, Massachusetts: The MIT Press, 2016. ISBN: 9780262337434.

[3] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: (1997). URL: https://papers.baulab.info/Hochreiter-1997.pdf.

[4] "IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Fast Basic Service Set (BSS) Transition". In: *IEEE Std 802.11r-2008 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008)* (July 2008). Conference Name: IEEE Std 802.11r-2008 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008), pages 1–126. DOI: 10.1109/IEEESTD.2008.4573292.

[5] "IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 1: Radio Resource Measurement of Wireless LANs". In: *IEEE Std 802.11k-2008 (Amendment to IEEE Std 802.11-2007)* (June 2008). Conference Name: IEEE Std 802.11k-2008 (Amendment to IEEE Std 802.11-2007), pages 1–244. DOI: 10.1109/IEEESTD.2008.4544755.

[6] *Indoor Location & Navigation | Kaggle*. https://www.kaggle.com/competitions/indoor-location-navigation. (Visited on July 11, 2023).

[7] *indoor-location-navigation-20*. URL: https://github.com/location-competition/indoor-location-competition-20 (visited on July 31, 2023).

[8] *Indoor Navigation: Complete Data Understanding*. https://kaggle.com/code/andradaolteanu/indoor-navigation-complete-data-understanding. (Visited on Apr. 25, 2023).

[9] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. "Deep learning for time series classification: a review". en. In: *Data Mining and Knowledge Discovery* 33.4 (July 2019), pages 917–963. ISSN: 1384-5810, 1573-756X. DOI: 10.1007/s10618-019-00619-1. URL: https://link.springer.com/10.1007/s10618-019-00619-1 (visited on Aug. 6, 2023).

[10] *Kaggle: Your Home for Data Science*. https://www.kaggle.com/. (Visited on July 23, 2023).

[11] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. *On the Difficulty of Training Recurrent Neural Networks*. Feb. 2013. arXiv: 1211.5063 [cs]. (Visited on Aug. 17, 2023).

[12] Pratap S. Prasad and Prathima Agrawal. "Movement Prediction in Wireless Networks Using Mobility Traces". In: *2010 7th IEEE Consumer Communications and Networking Conference*. Jan. 2010, pages 1–5. DOI: 10.1109/CCNC.2010.5421613.

[13] L.R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (Feb. 1989), pages 257–286. ISSN: 00189219. DOI: 10.1109/5.18626. URL: http://ieeexplore.ieee.org/document/18626/ (visited on Aug. 6, 2023).

[14] Szymon Szott, Katarzyna Kosek-Szott, Piotr Gawłowicz, Jorge Torres Gómez, Boris Bellalta, Anatolij Zubow, and Falko Dressler. "Wi-Fi Meets ML: A Survey on Improving IEEE 802.11 Performance With Machine Learning". In: *IEEE Communications Surveys & Tutorials* 24.3 (2022). Conference Name: IEEE Communications Surveys & Tutorials, pages 1843–1893. ISSN: 1553-877X. DOI: 10.1109/COMST.2022.3179242.

Acronyms

AP	Access Point
ARIMA	Autoregressive Integrated Moving Average
BSSID	Basic Service Set Identifier
GRU	Gated Recurrent Unit
HMM	Hidden Markov Model
LSTM	Long Short-Term Memory
MAC	Media Access Control
MajorID	Major Identifier
MinorID	Minor Identifier
ML	Machine Learning
MLP	Multilayer Perceptron
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
RSSI	Received Signal Strength Indication
SSID	Service Set Identifier
TxPower	Transmission Power
UUID	Universally Unique Identifier
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network

Zusammenfassung

...

Eidesstattliche Erklärung

Hiermit versichere ich, dass meine Bachelor's thesis "Machine Learning-based User Movement Prediction in Layer 2 Networks" ("Vorhersage von Benutzerbewegungen in Layer 2 Netzwerken basierend auf Maschinellem Lernen") selbstständig verfasst wurde und dass keine anderen Quellen und Hilfsmittel als die angegebenen benutzt wurden. Diese Aussage trifft auch für alle Implementierungen und Dokumentationen im Rahmen dieses Projektes zu.

Potsdam, den 17. August 2023,

(Lina Wilske)