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 22, 2023

Abstract

- human movement not considered when roaming of mobile devices appears
- could lead to many unnecessary handovers and thus to bad performance
- this thesis proposes a machine learning-based approach to predict the Access Point (AP) a user is nearest to based on the Received Signal Strength Indication (RSSI) of the APs
- the chosen machine learning model is Long Short-Term Memory (LSTM) and is trained on real-world data from a dataset of a competition by Microsoft Research
- the dataset analysis showed that interpolation is necessary to ensure that Wireless Fidelity (Wi-Fi) and waypoint data together are used in the prediction
- many APs are deployed in the buildings of the dataset, which makes the prediction task hard
- one building and one floor were chosen to be the dataset for this thesis
- the evaluation of the model shows that the mode predicts the AP a user is nearest to with a top 3 accuracy of about 71%
- too many classes, which makes prediction too hard and top 3 accuracies too low
- with fewer classes, the model could predict better
- in the future, data explicitly generated for this purpose could be used to predict the top k access points so that the location of the access points is known and can be considered in prediction

Contents

1 Introduction

In large-scale Wi-Fi environments such as office buildings, shopping malls, and airports, where multiple 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 and further towards AP3, ideally, AP3 should not initiate the roaming process but instead AP2 and then AP3 is connected. 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. Hence, existing solutions often lead to frequent hand-offs, resulting in dropped connections and unsatisfied users.

This thesis will explore if a time series Machine Learning (ML) model can predict the nearest AP a station may connect to next. This nearest station needs to be in the top 3 of the predictions to be considered a correct prediction. Hence, this thesis needs data with Wi-Fi, waypoint of clients, and sensor data, e.g., acceleration. A time series ML model requires input time series data. There are two possible data sources: generate new or utilize existing data. Data generation needs a comprehensive plan for accounting data setup and collection. This process is time-consuming and needs a lot of planning and evaluation beforehand, which is not the focus of this thesis. Thus, we will utilize pre-existing data from a 2021 competition by Microsoft Research[6]. The data will be analyzed in chapter 3 to determine what parts of the data we will use for the ML model. Additionally, the data will be prepared for a time series ML model.

After that, we will discuss the suitability of some pre-selected time series ML models for the task in chapter 4. Due to many data, we will discover in the chapter 3, this thesis will implement, in chapter 5, a ML model, train and test it for one site and floor of the competition. Finally, in chapter 7, we will evaluate the model's performance and conclude if this prediction could be useful.

2 Background

- in the following, some special terms used in this thesis are explained here

2.1 Time Series Prediction

Time series data, comprising a sequence of data points ordered in time, represents a typical structure in many domains, including user mobility within a Wi-Fi network. Owing to its inherent temporal dependencies—where subsequent data points influence 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 LSTM and Gated Recurrent Unit (GRU). Each model is designed to capture and leverage temporal patterns within the data, predicting future trends based on historical observations.

2.2 Hyperparameter tuning

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, the number of window and batch size. 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. Hyperparameters can be tuned by e.g., random search, which can be done manually or with the use of libraries such as keras-tuner.

2.3 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.

2.4 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

2.5 Temporal Dependency Handling

Temporal dependency handling refers to the ability of a ML 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 company XYZ[10] recorded the data in shopping malls 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 five 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 the 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 Basic Service Set Identifier (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, 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, the 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. Also, we only have 2,027 waypoints for this floor but 1,862,044 lines of Wi-Fi data, as seen in table 3.1. The visualization of the waypoints can be seen in fig. 3.1.

Further human movement may have occurred between *TYPE_WAYPOINT* and *TYPE_WIFI* data. 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 three 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.

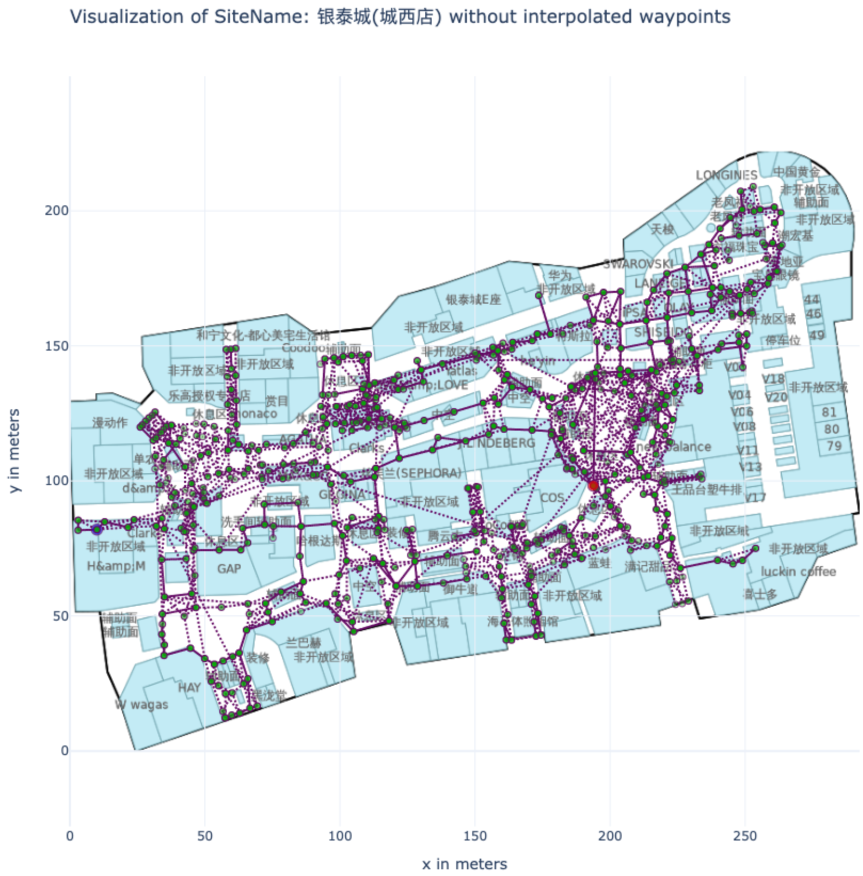

Figure 3.1: Visualization of the waypoints for SiteName: 银泰城(城西店)

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

- 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
```

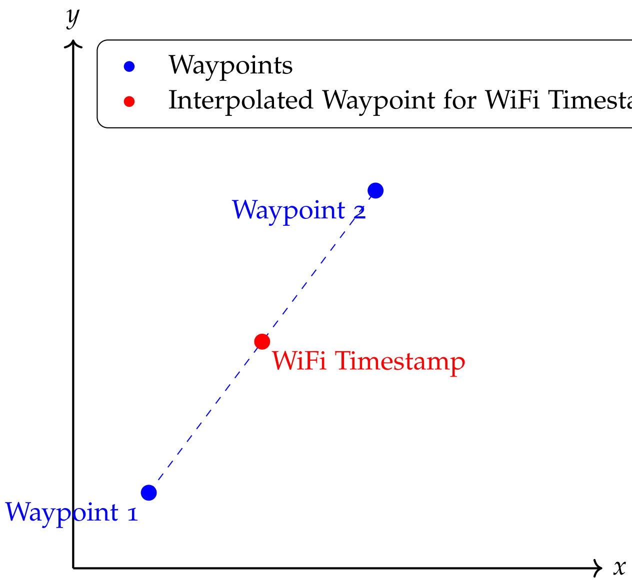

Figure 3.2: Visualization of linear interpolation for Wi-Fi timestamps based on given waypoints. The blue points represent the original waypoints, while the red points show the interpolated positions for specific Wi-Fi

```
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

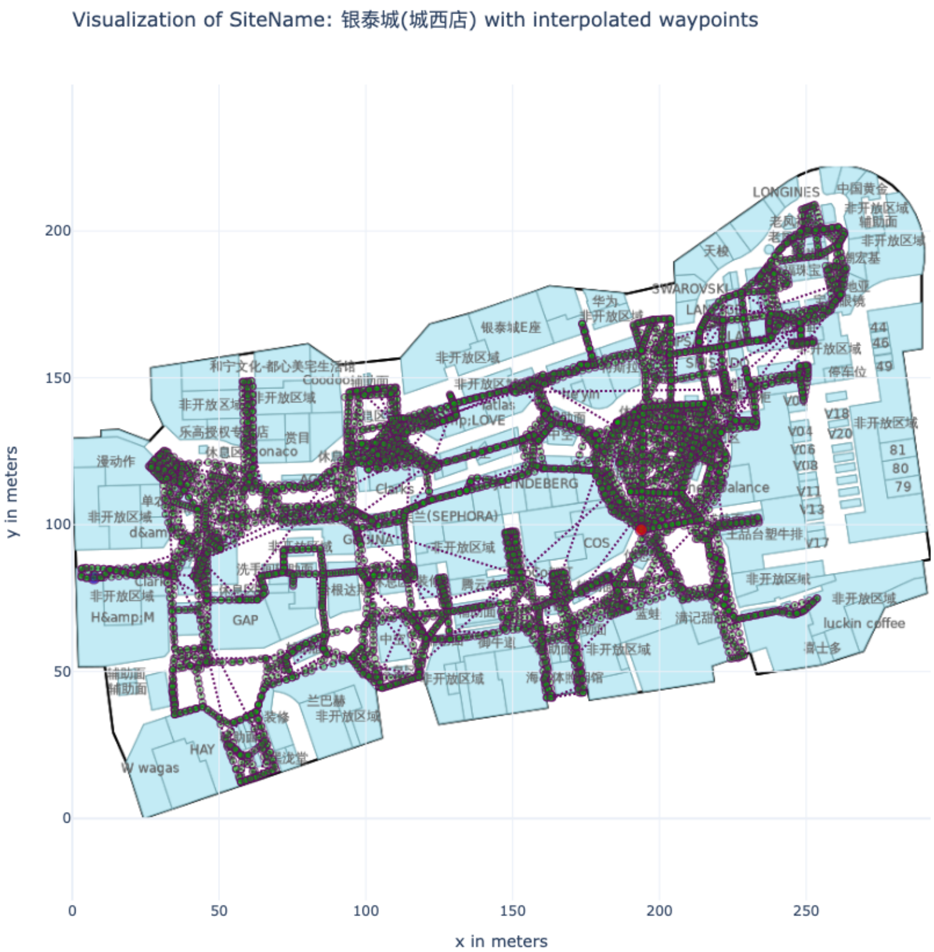

Figure 3.3: Visualization of the interpolated waypoints for SiteName: 银泰城(城西店)

4 Suitable Machine Learning Model

- As seen in the previous chapter, many 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 the problem as a 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) [13],

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, each 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[14]. HMMs are mainly 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 result from a sequence of hidden states, each producing an observable output according to a particular probability distribution.

4.1.3 RNN

RNN is an 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 Hochreiter and Schmidhuber introduced in 1997[3]. LSTMs were designed to combat the "vanishing gradient" problem in traditional RNNs. This problem made it difficult for other neural networks to learn from data where relevant events occurred with significant gaps between them. The key to the ability of the LSTMs 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 a suitable model for this task is even more critical. We will discuss the classification models by the following topics: Temporal Dependency Handling, Capacity and Complexity, Multivariate Data, Flexibility and Integration, and Regularization and Overfitting.

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[12].

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 effectively process each time step with multiple features.
- 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 advantageous when dealing with complex and varied data sources.

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

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 many classes. This problem demands a model that can efficiently capture intricate temporal patterns, scale in capacity, and handle multivariate data. 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

All the code for this implementation can be found in the GitHub repository [15].

5.1 Preprocessing

- Preprocessing with numpy and pandas
- Use data from chapter 3 for preprocessing
- Load data from files of floor with most files
- Create a target variable (based on RSSIs of BSSIDs)
- Normalize the data.
- Create sequences of data based on window_size variable.
- Encode the target variable, which is a variable with 4795, where 1 means the class is the nearest AP and 0 means the class is not the nearest AP, which results in a one-hot encoding.
- Split the data into training and testing set. (80/20)

5.2 LSTM Training and Testing

- Use Keras library for implementation [11]
- Model: LSTM layer, Dense layer with softmax activation

  Sequences are of length window_size for each entry in the dataset

  Inputs are (window_size, number_of_features (which are 6 + number of BSSIDs)), see fig. 5.1
- Generate predictions on the test set
- Get class with the highest probability as prediction
- Get top 3 predictions for the test set and check if the target variable is in the top 3 predictions.

5.3 Tuning model and hyperparameters

- try out different hyperparameters also in combination

  Number of units in the LSTM layer = {100, 150, 200, 350, 500, 1000, 2000}

  Number of epochs = {100, 200}

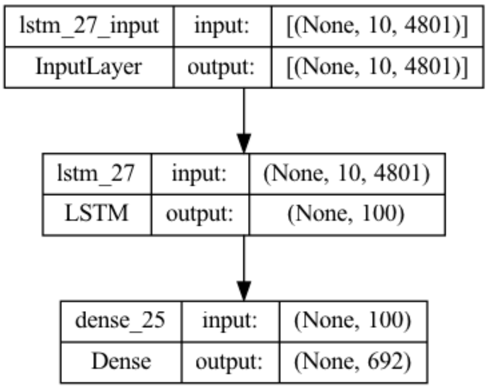

Figure 5.1: An example LSTM Network with Input, LSTM and Dense Layer with 4801 Features and window_size of 10.

batch size = {16, 32, 64}

window size = {3, 5, 10, 20}

6 Evaluation

- Top k: predicted class is within the top k predictions
- calculation for top k:

$$\frac{\binom{NUM\_CLASSES-1}{k-1}}{\binom{NUM\_CLASSES}{k}} = \frac{k}{NUM\_CLASSES}$$

- for the floor with most files: NUM_CLASSES = 4795
- Probabilities to pick one random BSSID, and it is the right one in top 3, 5 or 10, see fig. 6.1
- Accuracy of the model's prediction with a batch_size of 32, see fig. 6.2,fig. 6.3,fig. 6.4 and fig. 6.5
- Accuracy of the model's prediction with a batch_size of 16, see fig. 6.6
- Accuracy of the model's prediction with a batch_size of 64, see fig. 6.7

- Comparison with random selection of classes: lstm always better than random selection
- describing the plots
- best performance: 71%, see fig. 6.8
- Reasons:

  data contains too many classes, with fewer classes to predict the model could have performed better

  discussion could have missed a better model

  model very simple, could be improved by using other layers in between LSTM and Dense layers

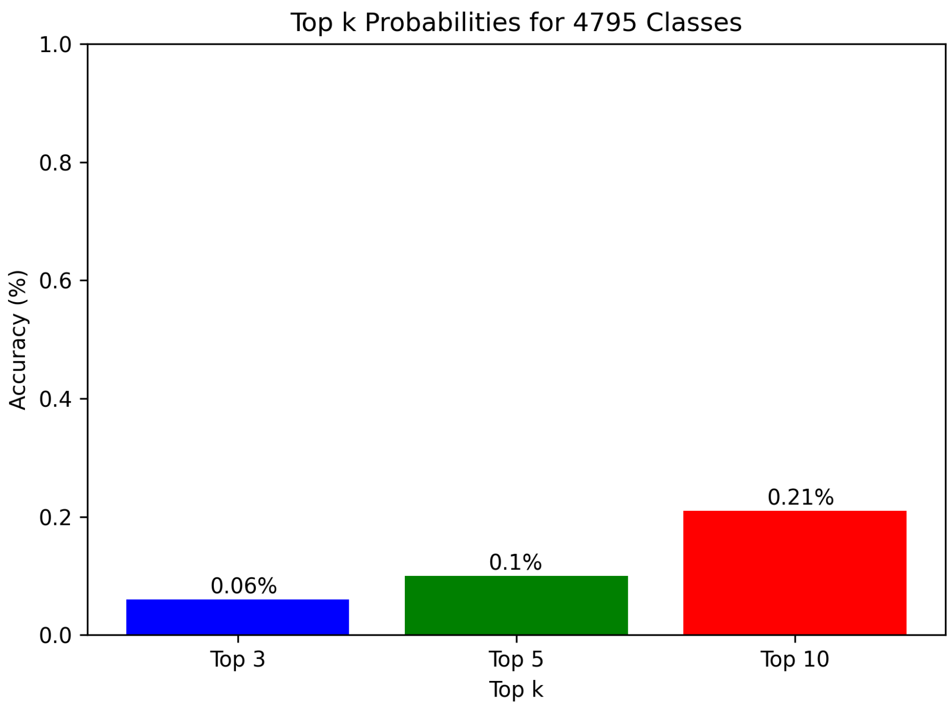

Figure 6.1: Probabilities that the predicted class falls within the top k randomly selected classes.

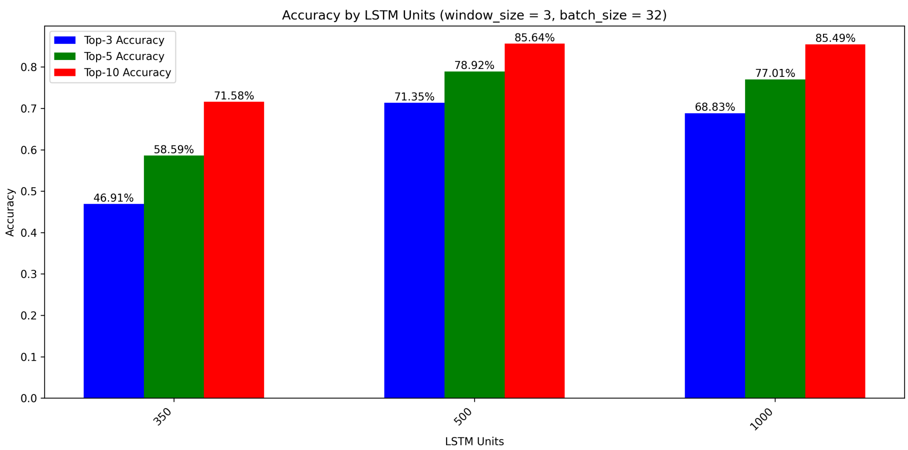

Figure 6.2: Accuracy of the model with window size of 3, batch size of 32 and 100, 500 and 1000 units in the LSTM layer.

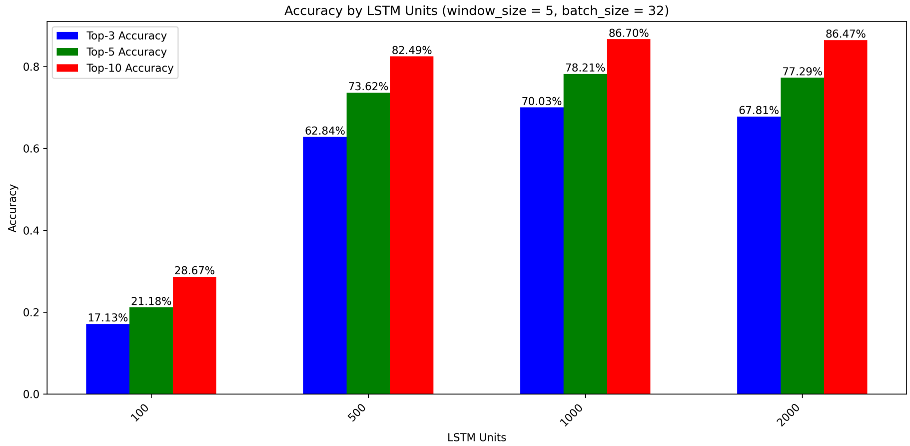

Figure 6.3: Accuracy of the model with window size of 5, batch size of 32 and 100, 500 and 1000 units in the LSTM layer.

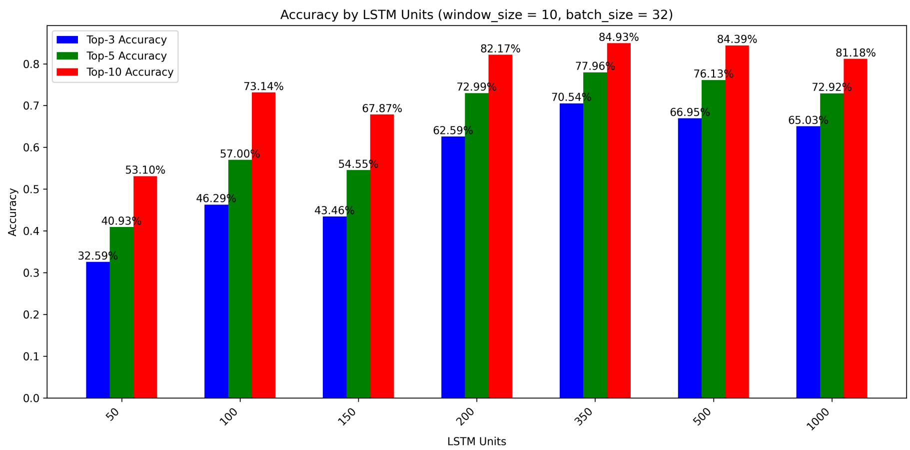

Figure 6.4: Accuracy of the model with window size of 10, batch size of 32 and 100, 500 and 1000 units in the LSTM layer.

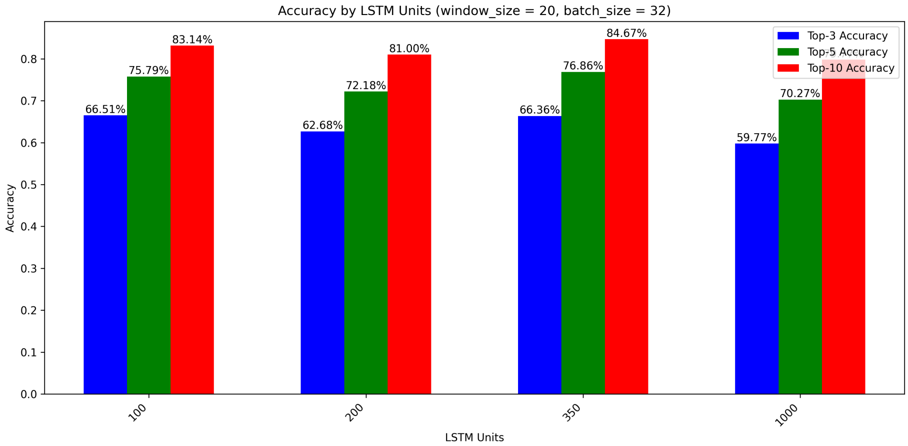

Figure 6.5: Accuracy of the model with window size of 20, batch size of 32 and 100, 500 and 1000 units in the LSTM layer.

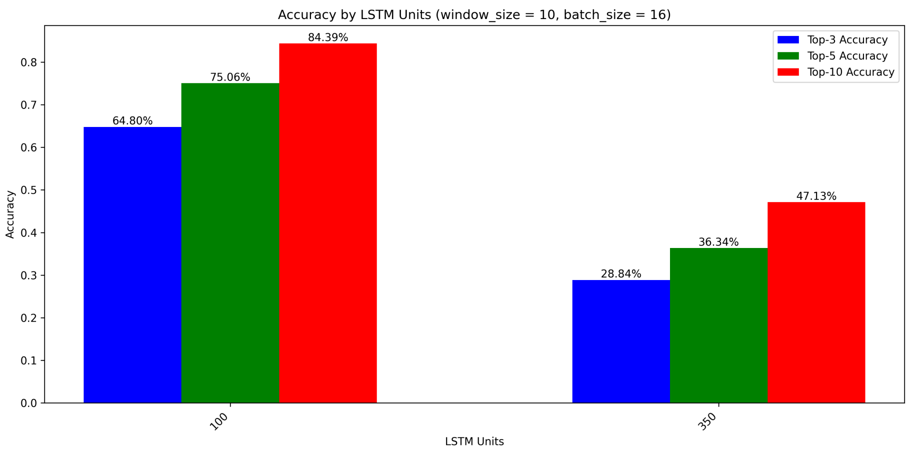

Figure 6.6: Accuracy of the model with window size of 10, batch size of 16 and 100, 500 and 1000 units in the LSTM layer.

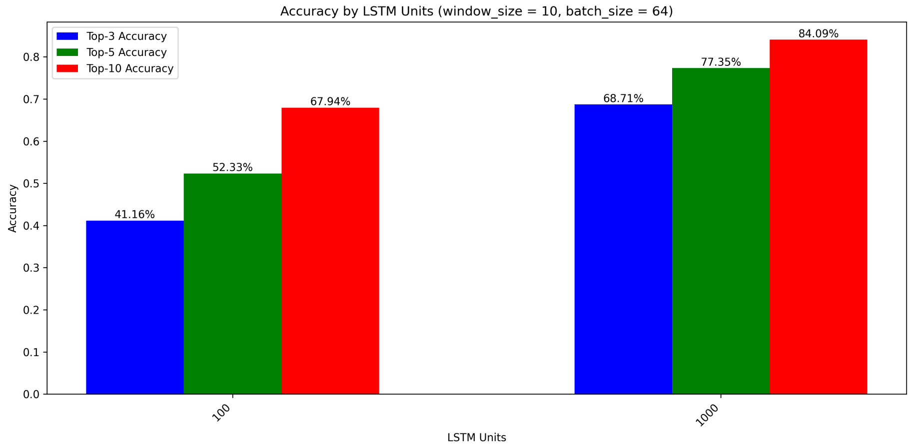

Figure 6.7: Accuracy of the model with window size of 10, batch size of 64 and 100, 500 and 1000 units in the LSTM layer.

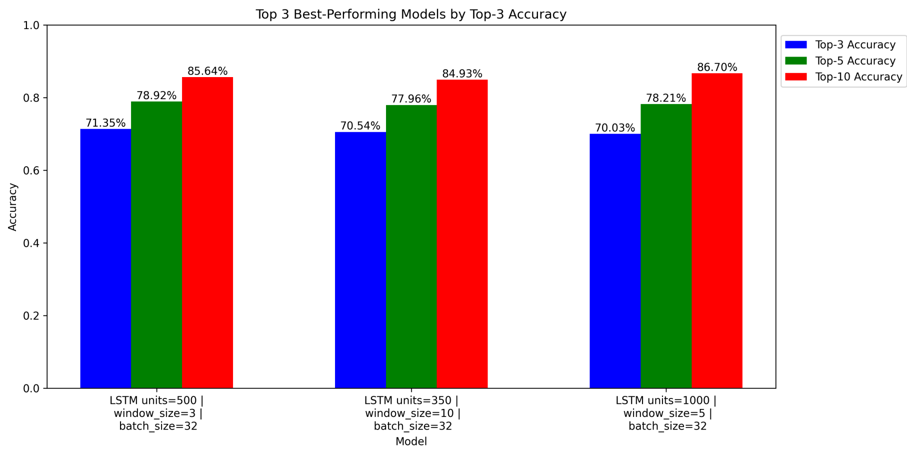

Figure 6.8: Accuracy of the three best performing models of the implementation

7 Conclusion

- indoor human movement prediction is a hard task with many classes
- although LSTM is the best choice for this task, the prediction accuracy is 70%
- with lesser classes, model could predict better, which could be a reason why ML model
- future work: use LSTM with fewer classes

  or generate data from mobile devices, so complete setup is known

  or generate data from APs to predict human movement

  could than be integrated in AP software like OpenWrt

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] *Keras: The high-level API for TensorFlow*. URL: https://www.tensorflow.org/guide/keras (visited on Aug. 18, 2023).

[12] 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).

[13] 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.

[14] 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).

[15] Lina Wilske. *GitHub Repository for bachelor thesis*. URL: https://github.com/linaScience/ba-implementation.

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

Zusammenfassung

To-do: translate english abstract to german

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 22. August 2023,

(Lina Wilske)