



Indoor Localization using transformers

SEP 769: Cyber-Physical Systems

Prepared by:

Group 13

Bassi, Ankur - 400347849

Patel, Harsh Sanjivkumar - 400354755

Patel, Tithi – 400490204

Contents

Introduction	3
Problem Statement/ Review / Background	3
Theory and Dataset.....	4
Implementation	6
Preprocessing:.....	6
Explanation of the Source code	9
Results and Discussion	10
Recommendations for Future work.....	13
References	14

Introduction

By utilizing machine learning strategies and the Microsoft Indoor Localization Dataset, the Indoor Localization and Navigation System project seeks to provide a high-precision indoor positioning and navigation solution. The goal of this project is to precisely pinpoint people's indoor positions utilizing Wi-Fi, Bluetooth, and inertial sensor data. The main task is to develop a sophisticated algorithmic framework that learns from different data sources over time to improve accuracy and flexibility. The ultimate objective is to offer customers seamless navigation direction with an intuitive interface that includes personalized visual, aural, or haptic cues.

A user-friendly, intuitive interface that smoothly combines with the indoor location and navigation system is another goal of the project. Users would be able to easily interact with the system using this interface, entering destinations and obtaining real-time instructions. Depending on the interests and needs of various users, the guidance mechanism may include a range of modalities, such as visual signals, audio instructions, or even haptic input.

The project is essentially a ground-breaking attempt to close the gap in precise indoor location and navigation. The project seeks to provide a ground-breaking solution that not only enhances our understanding of indoor settings but also refines it by utilizing the vast dataset and the power of machine learning.

Problem Statement/ Review / Background

There is a sizable challenge focusing on indoor location and navigation in the "Indoor Location Competition 2020" repository. Accurately determining people's whereabouts in complex indoor surroundings is the main challenge. Despite improvements in outside navigation, indoor navigation still poses a challenging challenge since it lacks GPS signals and has a variety of obstructions and signal interferences.

The difficulty also includes the requirement to create reliable algorithms that can successfully combine data from many sources, such as Wi-Fi, Bluetooth, and inertial sensors. It is quite difficult to successfully combine these data streams while taking into consideration their inherent noise and variability. Along with pinpointing interior locations, the competition is looking for solutions that can also offer trustworthy real-time navigation advice. To do this, it is necessary to solve concerns with algorithmic effectiveness, user-friendly interfaces, and flexibility to different indoor layouts.

The challenge is to develop novel strategies that go beyond conventional ways by fusing sensor data, machine learning, and data analysis. The main objective is to develop indoor positioning and navigation, which has applications in the retail, healthcare, and smart building management sectors. The problem challenges to develop innovative solutions that advance our knowledge of indoor places and increase how people move through and engage with them.

Our project's aim is to predict position of smartphones based on the real time sensor data, provided by indoor positioning technology company XYZ10 in partnership with Microsoft Research. The main task is to predict the floor and waypoint at specific time provided in the file named submission.csv from the existing data of site path file. This project can be implemented in real time in different malls as well as different industries. Here we have implemented the requirements on the dataset from the real malls of China.

Theory and Dataset

Initially we had data of 24 site(All of them are of the malls in China). Representation of the Floors are (B1,B2,F1,F2,F3,F4,F5,F6,F7,F8,F9) apart from floor we have few sensory data which are provide through Android(Accelerometer, Magnetic Field, Gyroscope, Wi-Fi Beacon) Utilizing all the above data we have to get following ground truth that is waypoint which represents the actual co-ordinates of the smartphone (X,Y).

indoor-location-competition-20

```
| README.md
| main.py                                //main function of the sample code
| compute_f.py                          //data processing functions
| io_f.py                               //data preprocessing functions
| visualize_f.py                        //visualization function
|
└── data                                //raw data from two sites
    ├── site1
    │   ├── B1                          //traces from one floor
    │   │   ├── path_data_files
    │   │   │   ├── 5dda14a2c5b77e0006b17533.txt //trace file
    │   │   │   └── ...
    │   │   └──
    │   │       ├── floor_image.png          //raster floor plan
    │   │       ├── floor_info.json           //floor size info
    │   │       └── geojson_map.json          //floor plan in vector format (GeoJSON)
    │   └──
    │       ├── F1
    │       └── ...
    └── site2
        └── ...
```

Following are the specifics of the information we have, the structure of the trace files, and the floor plan metadata:

1. Setup for Data Collection:

- An Android smartphone is used to collect data from indoor traces at two separate locations (malls).
- A site surveyor's body is in front of the smartphone, which is held flat.
- Several measurements are recorded using a sensor data recording app when the surveyor moves between points p1 and p2.

2. Sensor Data Types:

- IMUs (Inertial Measurement Units) are among the sensor data types that have been gathered. Readings: These include gyroscope and accelerometer readings. The smartphone's gyroscope detects rotation, while the accelerometer measures acceleration and orientation.
- Magnetometer for the Geomagnetic Field Readings: These readings can assist identify the Earth's magnetic field and offer information about it.

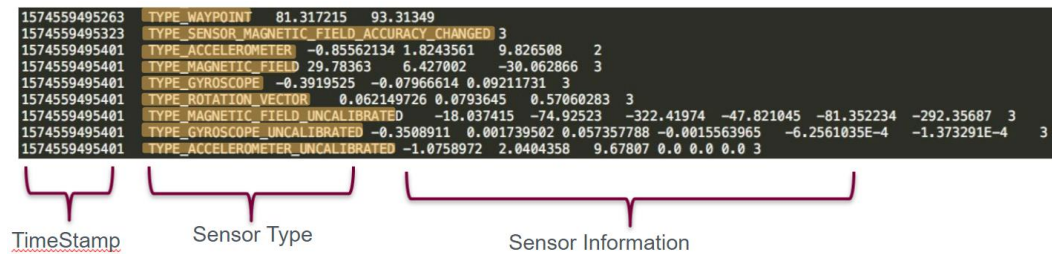


Fig. 1 Sensor information

3. Bluetooth and WiFi iBeacon Scanning:

- The smartphone searches for nearby iBeacon and WiFi devices. Using methods like fingerprinting, WiFi scanning can aid with interior placement, while iBeacon scanning can help locate specific areas.

4. Format of Trace Files:

- Each trace (*.txt) file represents a route taken between points p1 and p2 by the site surveyor.
- A structured format containing timestamped readings of the accelerometer, gyroscope, magnetometer, WiFi, and Bluetooth iBeacon data is most likely present in the trace file.
- Columns for timestamp, gyroscope, accelerometer, magnetometer, WiFi details (BSSID, RSSI, etc.), and iBeacon details (UUID, Major, Minor, RSSI, etc.) might be included in the format.

```
# startTime:1574559495255
# SiteID:5dd3d7732a57a34356595932 SiteName:杭州西溪银泰城 FloorID:5dd3d7732a57a34356595934 FloorName:F1
# Brand:OPPO Model:PBCM10 AndroidName:8.1.0 APILevel:27
# type:1 name:BMI160 Accelerometer version:2062600 vendor:BOSCH resolution:0.0023956299 power:0.18 maximumRange:39.22661
# type:4 name:BMI160 Gyroscope version:2062600 vendor:BOSCH resolution:0.0010681152 power:0.9 maximumRange:34.906586
# type:2 name:AK09911 Magnetometer version:1 vendor:AKM resolution:0.5996704 power:2.4 maximumRange:4911.9995
# type:35 name:BMI160 Accelerometer Uncalibrated version:2062600 vendor:BOSCH resolution:0.0023956299 power:0.18 maximumRange:39.22661
# type:16 name:BMI160 Gyroscope Uncalibrated version:2062600 vendor:BOSCH resolution:0.0010681152 power:0.9 maximumRange:34.906586
# type:14 name:AK09911 Magnetometer Uncalibrated version:1 vendor:AKM resolution:0.5996704 power:2.4 maximumRange:4911.9995
# VersionName:v20191105-nightly-16-gcd7805b VersionCode:403
```

Fig. 2 Meta info

5. Floor Plan Metadata:

The following floor plan metadata are given for each floor:

- **Raster Image:** A representation of the floor plan in the form of an image, most often in a graphic format (such as PNG or JPG).
- **Size:** The width and height of the floor plan picture.
- **GeoJSON:** A file format that uses JSON to store geographic information. It could specify the floor plan's geometrical characteristics, features, and limits in space.

IMU (accelerometer, gyroscope, and magnetometer) and magnetometer sensor readings from the smartphone are recorded as part of the data collecting process, along with searching for nearby WiFi and Bluetooth iBeacon devices. Each trace file includes a precise timestamped readout of the path between points p1 and p2. Additionally, each level has access to floor plan metadata, including photos and geographic data, to help with spatial context.

Implementation

Preprocessing:

Creating Features:

Orientation Estimation: IMU data includes information from accelerometers, gyroscopes, and often magnetometers. This information can be used to estimate the orientation of a device (e.g., a smartphone or wearable) relative to the Earth's surface. Knowing the device's orientation helps in understanding how it's moving through space.

Motion Tracking: By analyzing changes in acceleration and rotational rate from the IMU, you can infer how the device (and presumably the person carrying it) is moving. This includes understanding whether the device is stationary, walking, turning, climbing stairs, etc.

Enhancing Other Sensors: IMU data can be fused with other sensors (such as Wi-Fi or Bluetooth signals) to improve positioning accuracy. For example, Wi-Fi might provide a rough position fix, and IMU data could be used to refine that position based on detected movements

1. **Step Information:** The code uses accelerometer data and additional sensors to detect individual steps and calculate stride lengths and headings(direction of movement) for each step.
2. **Cumulative Distance:** The cumulative walking distance is calculated as the sum of stride lengths over time.
3. **Relative Positions:** The relative positions in X and Y directions are calculated using the headings and stride lengths, and the cumulative sum represents the total displacement in those directions.

Categorical Features available	Continuous Features available
Site ID	Wifi Signal
Wi-Fi ID	Beacon Signal
	Accelerometer
	Gyroscope, Rotation
	Walking Length
	Magnetometer

Table 1. Features available

There are more than 100 Wi-Fi signals associated with each path. We can extract the top 40 most reliable signals. Leading to 40 Wi-Fi ids and 40 signal strength.

- **Dimensionality Reduction:** Wi-Fi scans can detect many networks, especially in dense urban areas. By selecting the top 40 signals, the system focuses on the most relevant information, reducing the dimensionality of the data.
- **Noise Reduction:** Weaker signals, which are ignored by taking only the top 40, may come from distant access points and contribute noise rather than useful information.
- **Alignment with Human Movement:** Considering the strongest signals might align better with human movements in indoor environments, reflecting the actual paths that people are likely to take.

[illegible]

Fig. 3 Preprocessing

Here, we are considering wifi id's also because later with the help of neural network we will make embeddings that will identify some interdependence between each wifi.

Features for IMU Statistics:

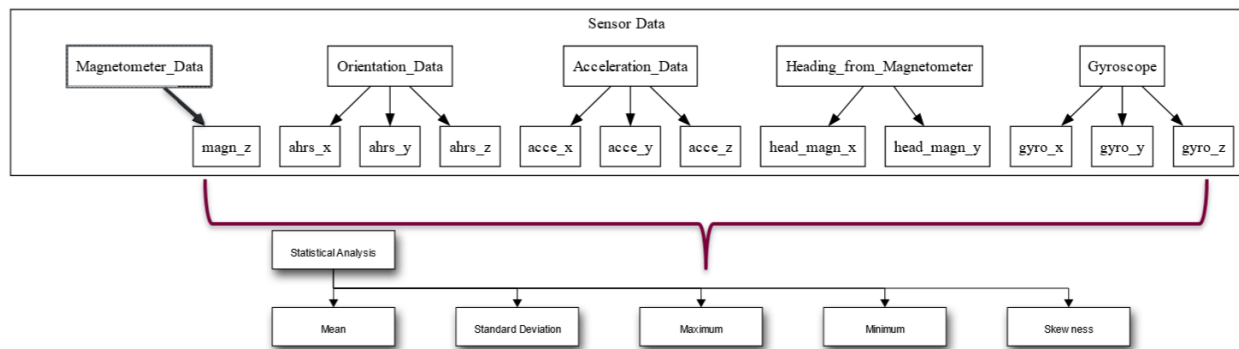


Fig. 4 Features for IMU Statistics

Feature Extraction: By calculating these statistical points, the code is extracting features that capture essential characteristics of the sensor data. This summarization can reveal underlying patterns and trends.

Dimensionality Reduction: Instead of using raw sensor readings, which can be numerous and noisy, summarizing the data with statistical measures reduces the dimensionality, making analysis and modeling more tractable.

Relative position and Walking Length:

The relative positions represent the cumulative displacement of an object from its starting point, measured in the x and y directions. The relative position is calculated in both the x and y coordinates. While working with different preprocessing here it would be necessary to emphasize on one of the important preprocessing steps that is Relative position.

A mall or any indoor unit might have different entrance or have user switched on the navigation application at different location hence the displacement to the initial position will be calculated in terms of x, y co-ordinates.

Basically, what function does is takes input data related to the walking path, calculates various step related information heading length and generates the sequence representing time and displacement and its cumulative position. as Discussed in the earlier slides now we have 146 includes site features, 80 wi-fi feature, 60 IMU features, 4 Relative position and 1 walking length features.

Explanation of the Source code

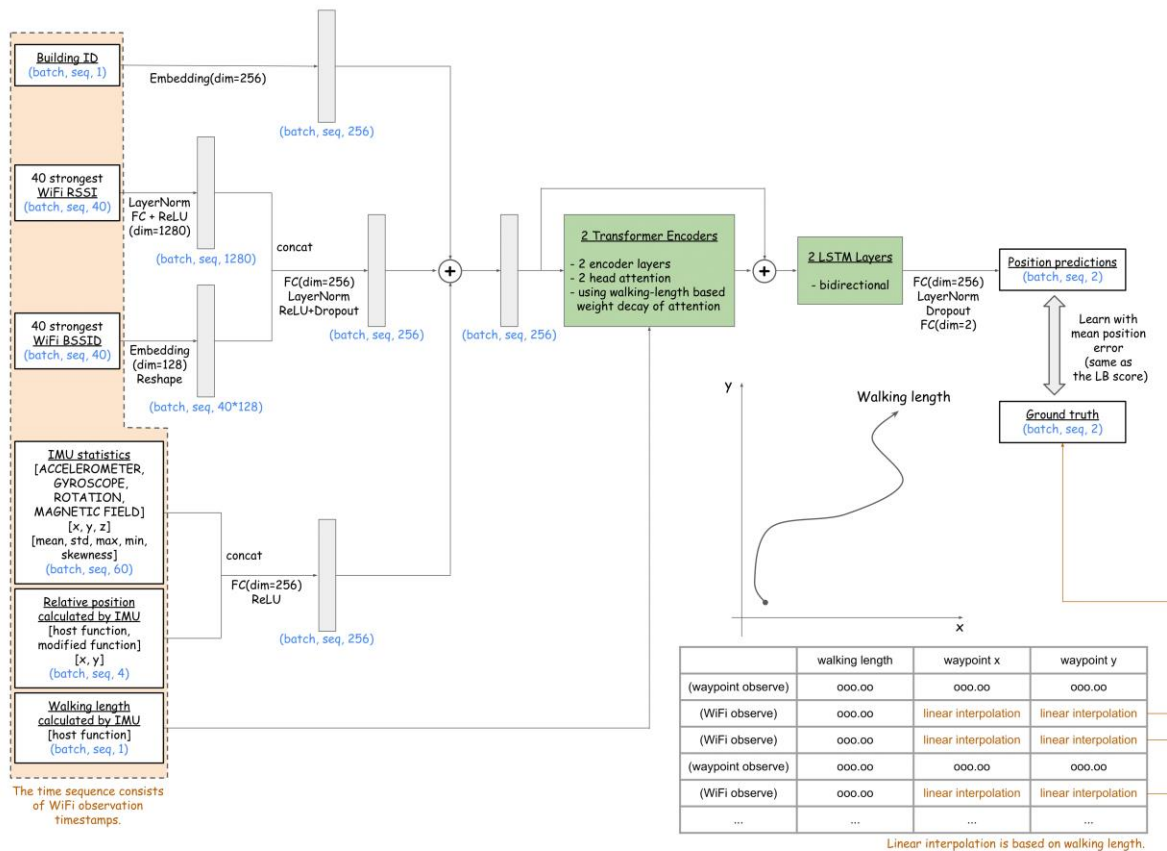


Fig. 5 Architecture overview

The information given describes how various data components are embedded and processed to provide positional coordinates in the end. Let's dissect and elaborate on each action:

- 1. Embeddings of site IDs:** Specific locations are represented by Site IDs, which are changed into 256-dimensional embeddings. With the help of this conversion, it will be easier to understand the intricate connections between distinct sites.
- 2. Embeddings for Wi-Fi ID:** Wi-Fi IDs are transformed into embeddings of 128 dimensions. The model can identify patterns and relationships in the Wi-Fi connectivity data thanks to these embeddings.
- 3. Processing the Most Powerful Wi-Fi Signals:** A fully linked neural network receives the 40 strongest Wi-Fi signals. ReLU activation functions are included into this network, resulting in a final output of 1280 neurons.
- 4. Fully Connected Layers and Concatenation:** The Wi-Fi embeddings are combined with the 1280 output neurons from the preceding stage. Following the passage of this combined data via completely linked layers, 256 neurons are produced.

5. Dimensions and Relative Position of the IMU: The four relative positional values (explained above) are concatenated with the 60 dimensions of the Inertial Measurement Unit (IMU) data. A 256-dimensional output is produced by feeding this combined data into a dense layer that is completely linked.

6. Architecture for Combinations and Transformers: A single 256-dimensional output tensor is produced by combining the three 256-dimensional output tensors that were formed in the earlier phases. Then, a transformer architecture with two encoder layers and two multi-head attention layers processes this tensor. This process assists in contextualizing.

7. Connection to LSTM: The original input data is combined with the output from the transformer design. Then, two bidirectional Long Short-Term Memory (LSTM) layers receive this merged data. By doing this, the model is guaranteed to represent underlying relationships and sequential dependencies between features.

8. How to Determine Positional Coordinates: Positional coordinates ('x' and 'y') are the output of the LSTM layers. According to the input data, which includes site embeddings, Wi-Fi signals, and IMU data, these coordinates indicate the estimated position.

IMU dimensions and relative position values are combined, Wi-Fi embeddings are combined with signal data, a transformer architecture is used, LSTM layers are integrated for sequential dependencies, and finally positional coordinates are determined. To estimate the precise "x" and "y" coordinates, which reflect the location under consideration, this comprehensive technique makes use of a number of data sources and neural network components.

Results and Discussion

Loss Function:

Positional Difference: The algorithm's learning process revolves around calculating the positional difference between the anticipated coordinates (x, y) and the actual ground truth coordinates (x, y) for a particular case. This serves as a key gauge of how well the model's predictions correspond to reality. The squared difference is employed, which accentuates greater differences and penalizes them more severely, helping to provide a clearer picture of the faults.

Aggregation: The technique takes the squared positional differences among a batch of instances and adds them up rather than just using them as individual examples. The squared difference between anticipated and actual coordinates is determined for each sample in the batch, and each squared difference is then added up. This total amount, which includes both the size and the direction of mistakes, is a depiction of the cumulative positional disparity for the whole batch.

Normalization: The loss measure is normalized to ensure that it accurately reflects the average mistake throughout the whole batch. Divided by the total number of cases in the batch is the aggregated sum of squared differences. By normalizing the error metric, which makes it independent of batch size and enables direct comparisons between situations, the error metric is made invariant to batch size.

Final Loss: The batch's final loss is calculated using the normalized aggregate. The average positional difference between anticipated and actual coordinates, scaled by the number of instances, is represented by this value. It provides a clear comparison of model performance over training iterations or models, and it captures how well the model is doing in terms of forecasting the right places.

Hyperparameters

The following are the parameters that can be tuned:

- Site id Embeddings(S_emb)
- Wifi Id Embeddings(W_emb)
- Number of features to pass to transformers(ninp)
- Number of Attention Heads(nAH)
- Number of hidden units in transformers feed-forward neural network(nhid)
- Number of Epochs

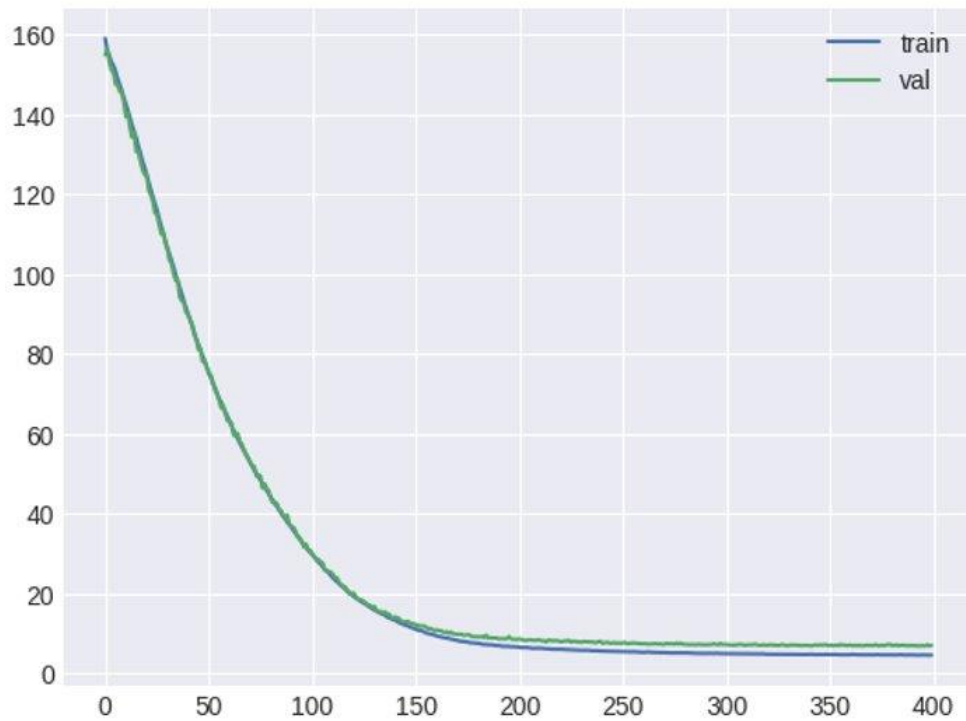


Fig. 6 Loss Graph

S_emb	W_emb	ninp	nAH	nhid	Epochs	Validation Loss
256	128	256	2	512	10	170
512	256	256	2	512	400	7.8
256	128	512	4	512	400	8.2
256	128	512	2	512	400	7.1

Fig. 7 Results

In the first configuration, the model embeds Wi-Fi IDs in a 128-dimensional array and site IDs in a 256-dimensional array. The model uses two multi-head attention layers, and the input data has a size of 256. The model architecture's hidden dimension is set at 512. The validation loss is reached after 10 training epochs, and it is 170.

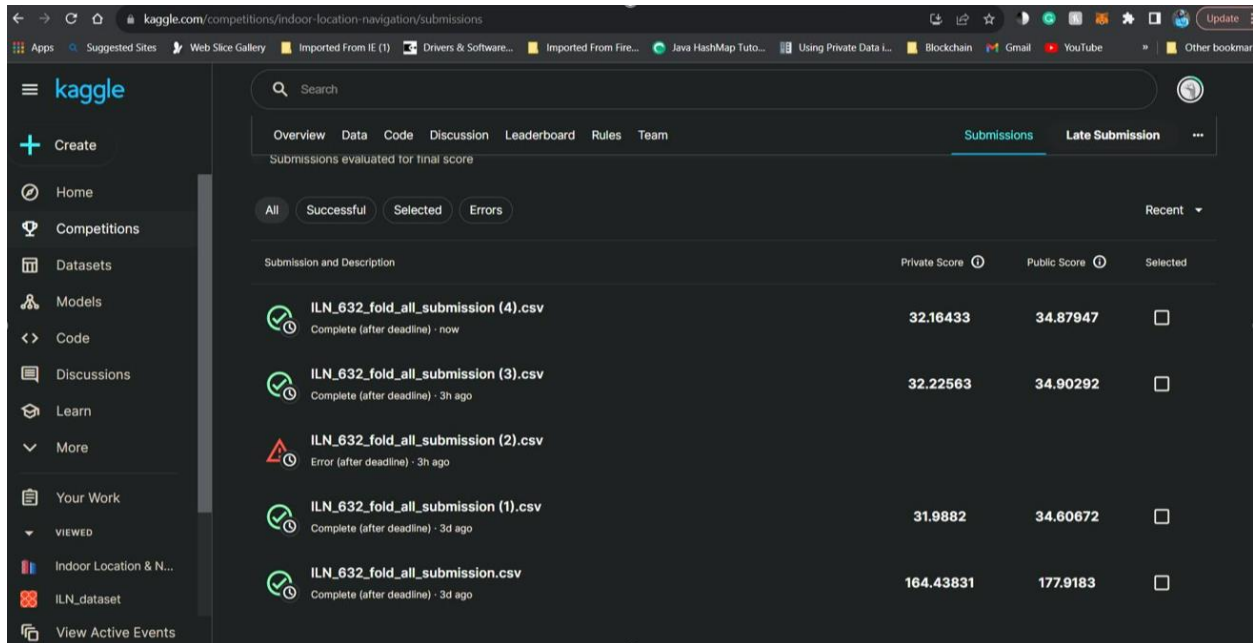
In the second setup, site IDs are embedded across a larger 512-dimensional space, while Wi-Fi IDs are embedded over a smaller 256-dimensional space. There are two multi-head attention layers used, and the input data still has a dimension of 256. The hidden dimension stays at 512 all the time. As a result of using a longer training period of 400 epochs, the validation loss dramatically decreased to 7.8.

In third configuration, the model returns to embeddings of 128 dimensions for Wi-Fi IDs and 256 dimensions for site IDs. The number of multi-head attention layers increased to 4, and the input dimension increased to 512. Once more, the secret dimension is set to 512. 400 training epochs are used, resulting in a validation loss of 8.2.

In fourth configuration, like the previous configuration, this one keeps site IDs embedded in 256 dimensions and Wi-Fi IDs in 128 dimensions. The input dimension stays at 512, and there are 2 levels with multi-head attention. Once more, the secret dimension is set to 512. Training involves 400 epochs, which results in a 7.1 reduced validation loss.

In conclusion, the data highlights the results of several model configurations, showing how varied selections in embedding dimensions, input dimension, multi-head attention layers, and training epochs affect the model's performance. It is clear that altering these parameters can result in significant variations in the validation loss, emphasizing the significance of hyperparameter tweaking and experimentation to optimize the model for the best outcomes.

Recommendations for Future work








Submission and Description	Private Score	Public Score	Selected
 ILN_632_fold_all_submission (4).csv Complete (after deadline) · now	32.16433	34.87947	<input type="checkbox"/>
 ILN_632_fold_all_submission (3).csv Complete (after deadline) · 3h ago	32.22563	34.90292	<input type="checkbox"/>
 ILN_632_fold_all_submission (2).csv Error (after deadline) · 3h ago			
 ILN_632_fold_all_submission (1).csv Complete (after deadline) · 3d ago	31.9882	34.60672	<input type="checkbox"/>
 ILN_632_fold_all_submission.csv Complete (after deadline) · 3d ago	164.43831	177.9183	<input type="checkbox"/>

Fig. 7 Kaggle Score

Our performance on Kaggle, as shown by our score of about 30 (a significant improvement from our previous score of 164), can be regarded as quite commendable, especially when taking into account the resource limitations we worked within and the extensive array of permutations and combinations we painstakingly explored.

By focusing on additional hyperparameter modification, we want to build on this success and continue boosting the effectiveness of our model. We want to conduct a number of thoughtful and planned experiments that will include the adjusting of numerous important variables. For instance, we plan to investigate the effects of including additional attention layers, which may help us uncover more complex links in the data. Additionally, we are prepared to increase the dimensionality of embeddings with the goal of extracting.

This process of fine-tuning will include a comprehensive investigation of many aspects of the model architecture. To enable the model to gather and analyze information more thoroughly, we will carefully study the consequences of increasing the size of fully linked layers. To achieve optimal convergence and reduce overfitting, the complex dance of hyperparameters—such as learning rates and regularization factors—must be painstakingly arranged.

Special Credits

We extend our deepest gratitude to the following individuals whose notebooks served as a boilerplate for our neural network architecture and data preprocessing code

- Housuke
- KhanhVD
- Kouki
- Minh Tri Phan
- Darich

References

- <https://github.com/location-competition/indoor-location-competition-20>
- <https://www.kaggle.com/c/indoor-location-navigation>
- <https://www.kaggle.com/code/horsek/iln-x-y-training-and-inference-part-no-pp>
- <https://developer.android.com/reference/android/hardware/SensorEvent.html#values>
- <https://www.kaggle.com/code/horsek/iln-preprocess-create-dataset-public/notebook>