WLAN Fingerprint-based Indoor Localization

Authors: Lluis Grifols, Marcel Romaní

Abstract: Modern day GPS technology still struggles when precisely locating individuals in indoor environments. The goal of this project is to train an algorithm to correctly locate an individual via WLAN fingerprinting. In this project we applied the following algorithms: Decision Tree, Random Forest, Extra Trees, K-NN, Nearest Centroid and MLP in order to predict Building, Floor, Latitude and Longitude variables.

I. INTRODUCTION

Many applications benefit from knowing the accurate location of their users. An example of such applications are medical alerts for seniors. These apps, when used in combination with devices such as smart watches, are a powerful healthcare tool as they are able to monitor blood pressure, heart-rate and even detect falls and are able to alert emergency services if they detect the user is in distress. However, GPS location may result inaccurate in indoor environments, hence a motivation to find solutions for indoor localization.

This project uses the RSSI (Received Signal Strength Indicator) from the detected WAP (Wireless Access Point) in order to predict the location of the user.

II. RELATED WORK

The proliferation of mobile phones with the capability to connect to WLAN in the late 00's opened the way for new indoor localization techniques and the number of publications about the topic increased ever since. This project uses the dataset provided by the Universitat Jaume 1 [1] [2].

III. DATA EXPLORATION AND PRE-PROCESSING

The information about the UJIIndoorLoc dataset allows us to learn this information:

1. The dataset covers three buildings of the Universitat Jaume I with four or five floors and a surface of almost 110000m^2.
2. The are a total of 21049 data points distributed as follows:
   (a) 19937 records for the training dataset.
   (b) 1111 records for the test dataset
3. There are a total of 529 attributes
   (a) **(001-520) - WAPXXX:** Intensity values for WAPXXX. Values range from -104 to 0 dBm. Values of 100 dBm correspond to measures that detected no signal.
   (b) **(521) - Longitude:** Longitude values in meters.
   (c) **(522) - Latitude:** Latitude values in meters.
   (d) **(523) - Floor:** Altitude in floors inside the building (0^{th} to 4^{th}).
   (e) **(524) - BuildingID:** ID to identify the building (0, 1 or 2).
   (f) **(525) - SpaceID:** Internal ID number to identify the Space (office, corridor, classroom) where the capture was taken
   (g) **(526) - RelativePosition:** Relative position with respect to the Space (1 - Inside, 2 - Outside in Front of the door).
   (h) **(527) - UserID:** User identifier.
   (i) **(528) - PhoneID:** Phone identifier.
   (j) **(529) - Timestamp:** UNIX Time when the capture was taken.

A. Pre-processing

Most of the pre-processing will take place in the "WAPXXX" attributes as they are the predictors we will work with. The first step was to take care of the artificial 100 dBm values, as well as those values lower than -100 dBm. To put values into perspective, 100 dBm is superior to the maximum power output of the High-frequency Active Auroral Research Program ($3.6 \cdot 10^6$ W) (most powerful shortwave station in 2012)[3], and -100 dBm is the mininimal received signal power of wireless network (10^{-13} W) [3]. Values of 100 dBm and those lower than -100 dBm where replaced to -100 dBm as we can assume it is a safe value to represent no signal received as well as to eliminate noise.

We conducted a search for missing values and found none. The attributes "SpaceID", "RelativePosition", "UserID", "PhoneID" and "Timestamp" where removed as we had no use for them in our models.

We also identified and removed those "WAPXXX" attributes that remained constant thoughout the training set or the test set, as information cannot be extracted or applied respectively. After this transformation a total of 311 "WAPXXX" attributes remain in out dataset.

B. Data Visualization

In Fig.1 we can see a the total number of appearances of each building in our dataset (both training and test sets). We can see there is an imbalance in the distribution, as building 2 represents almost half of the data points.

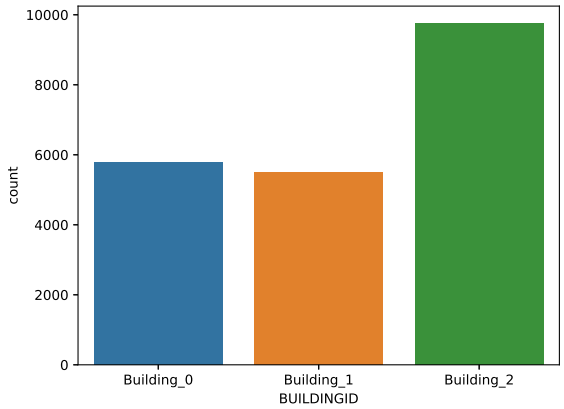

FIG. 1: Building distribution of the dataset.

Fig.2 shows us the distribution of floors among each building. We can notice that building 2 is the only one with a 4^{th} floor, as well as having the largest amount of data collected in the 3^{rd} floor.

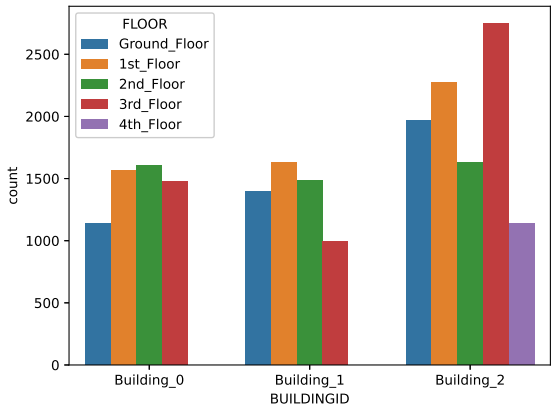

FIG. 2: Floor distribution of the dataset.

We can also differentiate between the training and test datasets as seen in Fig.3. In the training dataset the floor distribution is approximately uniform with the exception of the 4^{th} floor. For the test set we notice a large amount of data corresponding to the 1^{st} and 2^{nd} floors.

In order to explore the Latitude and Longitude attributes we decided for a 2D representation of Longitude vs. Latitude as they are coordinates of the earth's surface. In Fig.4 we can see how each data point assigned

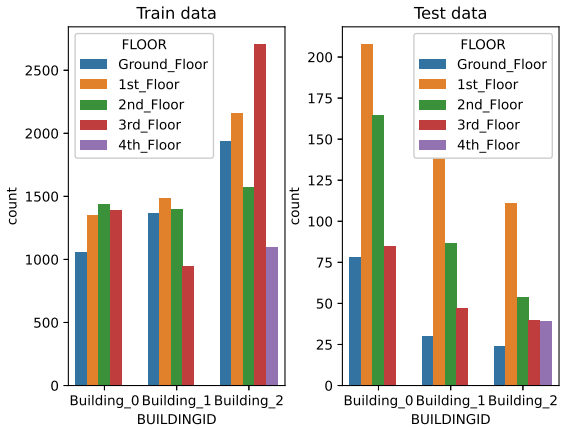

FIG. 3: Floor distribution of the Training and Test sets.

to a certain building is in fact inside the limits of said building. The Longitude-Latitude plot is superimposed to a real life satellite image of the buildings (as seen on Google Maps).

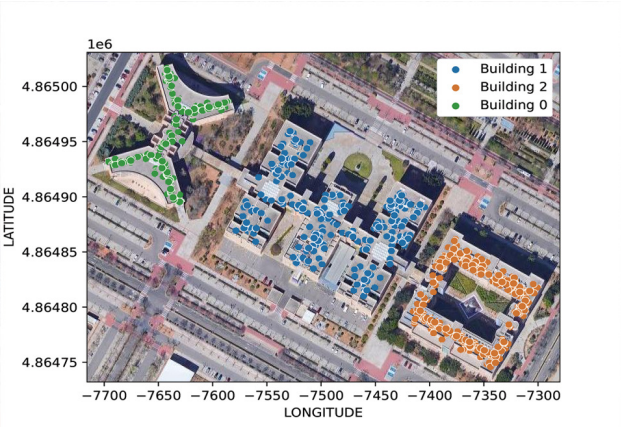

FIG. 4: Longitude and Latitude representation superimposed to real Google Maps image

By adding a vertical dimension to Fig.4 we obtain Fig.5. This plot gives us a very good representation of our dataset, as we are able to see Latitude, Longitude, Buildings and Floor all in one figure.

Under the assumption that all "WAPXXX" attributes are equal in importance, we explored all our "WAPXXX" attributes as a whole and see how the distribution of RSSI (Received Signal Strength Indicator) looked like. The distribution can be seen in Fig.6

IV. CLUSTERING

Now we are going to see how our data is clustered by only taking into account the independent variables, i.e. without using the information about building, floor or

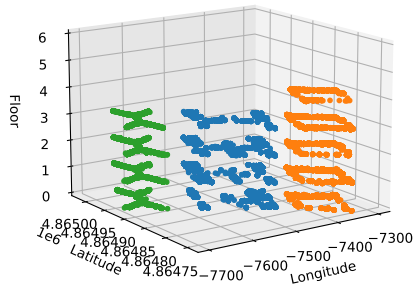

FIG. 5: 3D representation of the dataset.

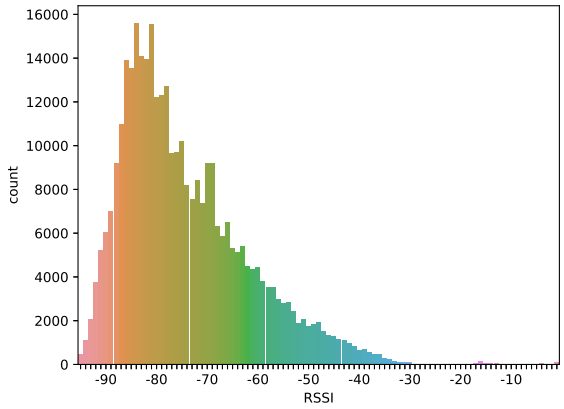

FIG. 6: RSSI frequency.

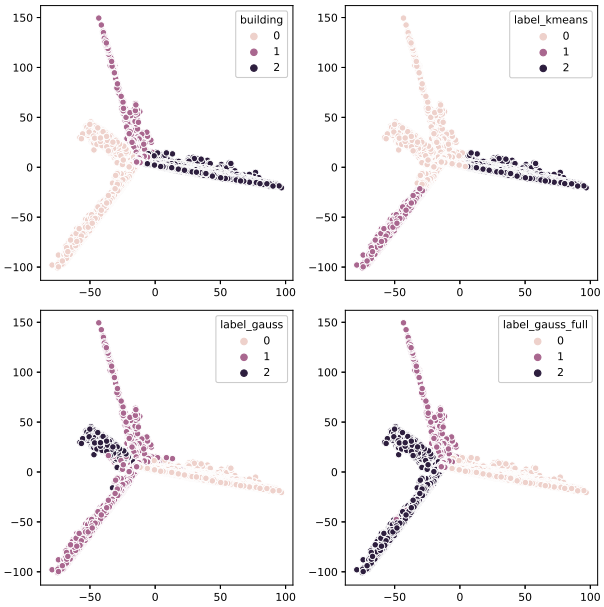

FIG. 7: RSSI frequency.

TABLE I: Clustering scores

	K-Means			GM-Full			GM-Diag		
k	**CH**	**Silh.**	**DB**	**CH**	**Silh.**	**DB**	**CH**	**Silh.**	**DB**
2	2131	0.102	2.89	1943	0.095	3.06	2022	0.097	2.99
3	1988	0.126	2.52	1714	0.111	2.83	1806	0.128	2.64
4	1999	0.133	2.05	1467	0.093	2.88	1378	0.054	3.44
5	1957	0.150	2.01	1328	0.094	2.71	1539	0.074	2.87
6	1747	0.154	1.94	1176	0.075	2.66	1443	0.054	2.68
7	1958	0.174	1.85	1461	0.138	2.24	1558	0.093	2.48
8	1910	0.191	1.73	1306	0.124	2.51	1413	-0.01	2.18
9	1742	0.187	1.74	1261	0.128	2.39	1360	0.004	2.11

coordinates.

First of all we were interested to see if imposing the number of clusters equal to 3 (the number of buildings) our data was separated in the same way. The following graph 7 shows the results using the K-Means, Gaussian Mixture with diagonal covariance and Gaussian Mixture with full covariance.

As we can see, the Gaussian Mixture with full covariance is able to capture much better the information about the 3 distinct buildings. Anyway, since we are working with unsupervised algorithms we don't expect a full distinction since there are other features, e.g. the floor, that could create differences in data that lead to other clusters.

Following with the clustering, we ran the three mentioned algorithms with varying number of clusters from 2 to 9 to see if there existed any grouping that we could interpret in some way. To evaluate the goodness of each model, we extracted the 3 principal scores, namely the Calinsky-Harabasz score, the Silhouette score and the Davies-Bouldin score. The results are shown in I.

As we can see, there isn't any particular configuration that is clearly a better clustering than all the others, even though the ones with lower k obtained a higher CH score. If we look at the GM with full covariance and $k = 3$, which we previously saw that had a similar labeling to the one corresponding to the different buildings, we see that the clustering scores do not particularly agree in that it's a clear winner. This shows that the data do not always look as we expect, but there are some underlying structures that do not match our divisions.

V. METHODOLOGY

A. Building and Floor Prediction

To predict the values of Building and Floor we used classification methods, as our target variable are of categorical type. In this section we used Decision Tree, Ran-

dom Forest, Extra Trees, K-Nearest Neighbours, Nearest Centroid and Multi-Layered Perceptron. We also performed a Principal Component Analysis and applied again the methods over the result.

In this section no scaling was applied to the data because all predictors share the same range and therefore are no differences between the scale of the variables.

Decision Tree. For this method we used default values, as using Random Forest would yield better results that a fine tuned Decision Tree.

Random Forest. For this section we used both a default Random Forest and a tuned Random Forest. Using the GridSearch function a combination of multiple parameters where tested. The parameters tested for the Random Forest in this section are:

1. max. depth = 100, 200, 300, None
2. min. samples leaf = 1, 2, 4
3. min. samples split = 2, 4, 8
4. class weight = balanced, balanced subsamples, None
5. n estimators = 100, 200, 300, 400, 500

The parameters that yield the best result for the Building model are: max.depth = 200, min. samples leaf = 1, min. sample split = 8, class weight = None and n estimators = 300.

For the Floor model the best parameters are: max.depth = 200, min. samples leaf = 1, min. sample split = 8, class weight = balanced and n estimators = 300.

Extra Trees. Being similar to Random Forests, we used the same set of parameters as Random Forests for the fine tuning, and obtained the same results as before.

K-Nearest Neighbours. For K-NN we tried different distance metrics and number of neighbours. The complete list of parameters tested is:

1. metric = Euclidean, Manhattan, Chebyshev, Minkowsky
2. n neighbors = 2, 4, 5, 8, 10, 15

Both Building and Floor models obtained the same results for the best parameters, these being: n neighbors = 5 and metric = Euclidean

Nearest Centroid. To tune Nearest Centroid we only used the metric parameter.

1. metric = Euclidean, Manhattan, Chebyshev, Minkowsky

Both Building and Floor models obtained metric = Chebyshev as the best metric

Multi-Layer Perceptron. We only used this method for the Floor model and after applying the PCA. We tested different depths and sizes as well as the alpha value. The values tested for this section are:

1. hidden layers = [2],[4],[6],[2,2],[4,4],[6,6]
2. alpha = 0, 0.001, 0.01, 0.1, 1, 10

The resulting best parameters are: hidden layer size = [4] and alpha = 0.

PCA A PCA was performed in order to reduce dimensionality. We kept the first 54 dimensions, with a cumulative explained variance of 90%.

We also tried to improve the model by creating a new target variable that combined both Building and Floor. In this part we used Random Forest, and then applied a PCA and Random Forest, Extra Trees and K-NN, while using the same hyperparameters that we obtained for Floor.

B. Longitude and Latitude Prediction

To predict the values of the Latitude and Longitude features we used regression algorithms of the same family as those used in classification in the previous section. Namely, we used Decision Tree Classifier, Random Forest Classifier and Mutilayer Perceptron. The regression of the two variables was done separately, but given the similarity of the inputs, the same set of hyperparameters was optimal for both models. We also opted to scale the variables to facilitate the learning process, specially in the case of the MLP.

Decision Tree and **Random Forest**. These two tree models were run with the default parameters, as in a first approach we wanted to compare their performance and chose the best one to fine tune in a following step.

Random Forest with Random Search. The space of parameters to look into had a total of 2420 different configurations, and 100 of them where run in a 3-fold cross validation in order to find the optimal one. The yielded best parameters were the following: $n\_estimators$=850, $min\_samples\_split$=10, $max\_features$='sqrt', max_depth=130, $min\_samples\_leaf$=1 and $bootstrap$=False.

Multi-Layer Perceptron. In this case we run a perceptron with 3 hidden layers of size 16. We added a L2 regularization term of $\alpha = 1$, and the optimization process was done with the *adam* optimizer with an initial learning rate of 0.0001. By using *early_stopping*, the intermediate models of the training process where validated against a 10% of the training data and the algorithm was set to finish when this validation didn't improve in a certain number of epochs.

VI. RESULTS

A. Building and Floor Prediction

In order to compare the results of different methods we are going to use the accuracy metric and the f1-score

metric. In addition, we have computed our own metric (named Custom error score). The formula for this error is as follows:

$$E = P_b \sum |B_p - B_t| + P_f \sum |F_p - F_t| \qquad (1)$$

where P_b is the penalty associated to Building, $P_b = 10$. P_f is the penalty associated to Floor, $P_f = 1$. B_p, B_t, F_p and F_p are the predicted building, the true building, the predicted floor and the true floor respectively.

The idea of this error is to penalize a wrong prediction by how far off it is. The reason behind this is that in the case of emergency services uses this model to locate someone inside a building because they need medical assistance, the further off the prediction is, it will take more time for the medical team to get to the correct location, and in medical emergencies time can be a crucial factor.

TABLE II: Building results

	Accuracy	**f1-score**	**Custom error score**
DT	0.97	0.97	420
RF	0.999	0.999	10
ET	1	1	0
KNN	0.996	0.996	50
NC	0.998	0.998	20
DT_pca	0.994	0.994	70
RF_pca	0.999	0.999	10
ET_pca	1	1	0
KNN_pca	0.999	0.999	10

In table II we can see the metrics obtained for the Building prediction. We can notice that Extra Trees gives the best performance both before and after the PCA. ExtraTrees gives us perfect performance with no missed predictions. Random Forest only misses one prediction both before and after the PCA, and KNN after the PCA also only one missed predition.

TABLE III: Confusion matrix for Building prediction using Extra Trees

	Building 0	Building 1	Building 2
Building 0	536	0	0
Building 1	0	307	0
Building 2	0	0	268

In table IV we can see the metrics obtained for the Floor prediction. We can see that the method that obtained the best scores for all three metrics is Random Forest with PCA, with 93% accuracy and a Custom error score of 96.

TABLE IV: Floor results

	Accuracy	**f1-score**	**Custom error score**
DT	0.82	0.81	239
RF	0.91	0.90	110
RF-Best	0.92	0.90	107
ET	0.91	0.91	109
KNN	0.92	0.92	105
NC	0.86	0.83	166
DT_pca	0.86	0.86	183
RF-Best_pca	0.93	0.92	96
ET_pca	0.92	0.92	104
KNN_pca	0.92	0.91	115
MLP_pca	0.91	0.88	119

TABLE V: Confusion matrix for Floor prediction using Random Forest + PCA

	Floor 0	Floor 1	Floor 2	Floor 3	Floor 4
Floor 0	114	10	8	0	0
Floor 1	7	431	22	2	0
Floor 2	2	6	291	7	0
Floor 3	0	0	9	162	1
Floor 4	1	1	0	3	34

In table VI we can find the results for the simultaneous prediction of Building and Floor. We can see that Random Forests with PCA produces a 94% accuracy model with only 85 Custom Error Score. This is the best model overall, as it precisely identifies all buildings and gives the best predictions for the floors.

TABLE VI: Building + Floor results

	Accuracy	**f1-score**	**Custom error score**
RF-Best	0.91	0.90	135
RF-Best_pca	0.94	0.93	85
ET_pca	0.93	0.92	99
KNN_pca	0.92	0.91	124

TABLE VII: Building Confusion matrix for Building+Floor prediction using Random Forest + PCA

	Building 0	Building 1	Building 2
Building 0	536	0	0
Building 1	0	307	0
Building 2	0	0	268

TABLE VIII: Floor Confusion matrix for Building+Floor prediction using Random Forest + PCA

	Floor 0	Floor 1	Floor 2	Floor 3	Floor 4
Floor 0	114	11	7	0	0
Floor 1	4	420	37	1	0
Floor 2	2	7	291	6	0
Floor 3	1	0	9	161	1
Floor 4	1	0	0	6	32

B. Longitude and Latitude Prediction

The predictions of the different regression models were evaluated using the Minimum Squared Error and the R^2 score. Given that our data was normalized, we will need to re-scale it back in order to interpret how good our predictions are by having distances back to *meters*.

Results can be seen in the following table.

TABLE IX: Latitude and Longitude

	Lat. MSE	**Lat. R2**	**Lon. MSE**	**Lon. R2**
RT-default	0.081	0.926	0.019	0.980
RF-default	0.040	0.964	0.012	0.988
RF-Best	0.019	0.983	0.006	0.993
MLP	0.071	0.936	0.033	0.965

As we can see, the algorithm with the best performance is the RF-Best, which corresponds to the Random Forest whose hyperparameters have been obtained using RandomSearch. The R^2 scores for the latitude and longitud features are 0.983 and 0.993, and the Mean Squared Error is 0.019 for the latitude and 0.006 for the longitude.

These values of the MSE can be interpreted by scaling back the data as follows:

$$MAE(m) = \sqrt{scale\_factor^2 * MSE} \tag{2}$$

The best results, corresponding to the RF-Best have

- Latitude Mean Absolute Error (m): 9.28m
- Longitude Mean Absolute Error (m): 9.85m

The following figures show where fall the predicted points (in blue) for each model compared to the real ones (in orange).

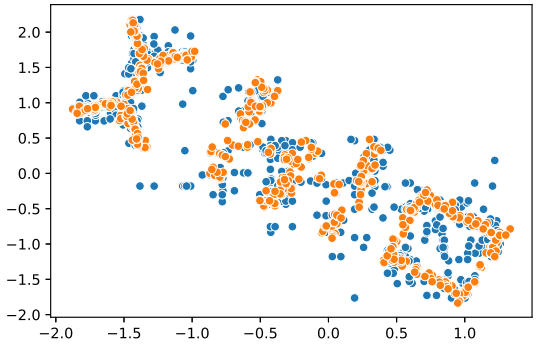

FIG. 8: Localization DT

As we can see in the plot above, the DT does a pretty good job at locating the source, but there are still many points that fall outside the buildings.

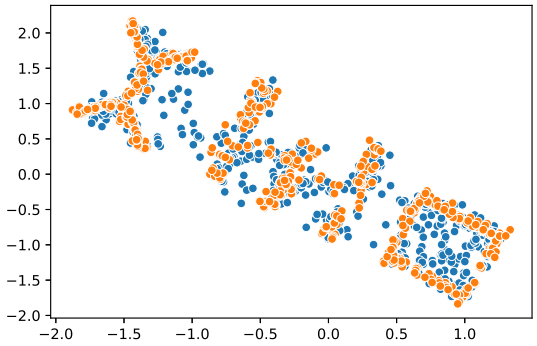

FIG. 9: Localization RF

The RF obtains better results, now there isn't almost any point that falls far away from the building. Now one of the biggest issues appears in the rightmost building, where most of the points fall in the middle.

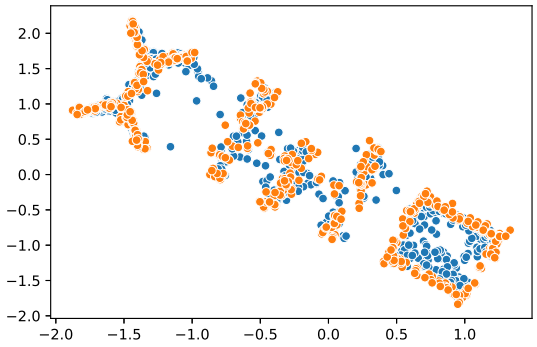

FIG. 10: Localization RF-Best

Clearly the model in the graph on 10 is the best at finding the location. In the leftmost building almost all the points are correctly within the bounds, and the rightmost has improved a little bit.

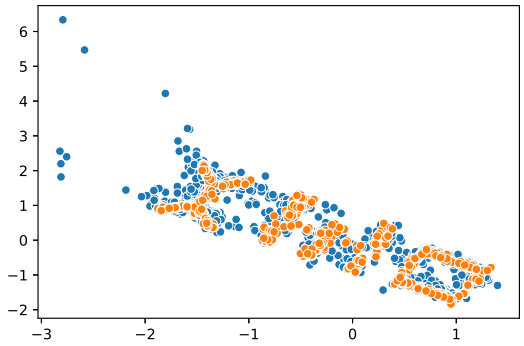

FIG. 11: Localization MLP

Finally, the MLP model is the worst performing (as we had anticipated in the previous table) mostly because of some points that have been cast away. This probably is due to some overfitting, that has lead to a very complex model with big variability. We should note, though, that predictions in the rightmost building are pretty good.

As we have seen, the building where the models have more issues with is the rightmost one. It is probably due to the big open space that lies within the walls, which may cause some interference or mixing of the WLAN signal.

VII. CONCLUSION

In this project we have tested multiple models with various parameter configurations in order to optimize the results, from which we have selected Random Forest + PCA for Building+Floor classification, and Random Forest for Latitude and Longitude regression. Even though results were pretty good and there's little room to improve, next steps would include further testing in the effect of the value we set as the lowest detectable intensity, and it would also be good to try other models such as Support Vector Machines, which could result useful to improve the accuracy of the regression. Overall we are confident in our models and further confirm that WLAN fingerprinting is a viable method for indoor localization.

[1] *UJIIndoorLoc Data Set* https://archive.ics.uci.edu/ml/datasets/ujiindoorloc

[2] Joaquín Torres-Sospedra, et al. *UJIIndoorLoc: A New Multi-building and Multi-floor Database for WLAN Fingerprint-based Indoor Localization Problems* (In Proceedings of the Fifth International Conference on Indoor Positioning and Indoor Navigation, 2014).

[3] https://en.wikipedia.org/wiki/DBm