Data Challenge: Geolocation of Connected Devices

Valentin Charvet, Maxence Monfort

November 30, 2018

This document presents the results of a data-challenge as part of IA317 Large Scale Machine Learning course from Artificial Intelligence Master (Télécom ParisTech and ENSTA ParisTech). The study consists in geolocating connected devices based on the send signals. [1]

1 Introduction

Along with the rise of the number of connected devices during the last decade (mobile phones, IOT etc...), questions have been raised with report to the geolocalization of those, that is finding accurate coordinates. Such information can be very useful for many applications and thus lead to many interests in industries and services. As a consequence, the research community has produced a number of publications as shown in survey [5].

The problem at stake is the following: given a network of M base stations (or antennas) and a set of N messages received by the stations, how can one retrieve the latitude and longitude of each emitting device? Let's precise that each message can be received by several antennas and can be in motion meaning we want to predict the device's coordinates at the time of message emission.

However, many publications related to this problem restrict their study to the localization of objects in a closed environment like a house for which the receiving devices are relatively close to the emitter (less than 100 meters) as in [4]. Unfortunately, those don't apply to our dataset since our distance is centered around 6.6 km with a standard deviation of 13.4, see Fig 1.

Moreover, the high number of samples as well as the uncertainty of the measurements has led us to think a statistical method would be appropriate. Consequently, we based our predictions on a machine learning two-target regression model. In section 2.2 we describe the process of feature engineering necessary to obtain an interpretable feature matrix. We also describe methods and approximations we have made to estimate distances between devices and antennas. In section 3 we show the empirical results of our studies and then in 4 we conclude.

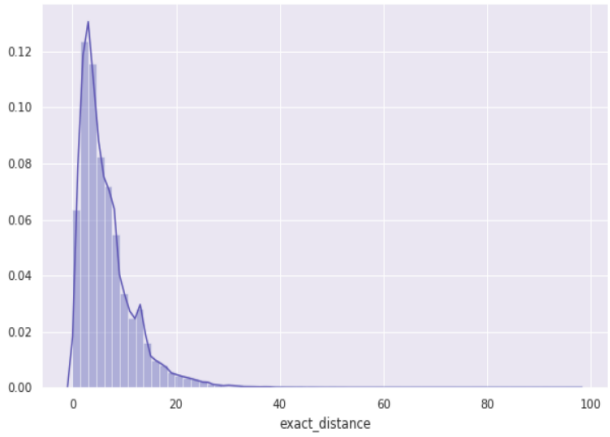

Figure 1: Distribution of distances smaller than 100km

[1] Full code available at https://github.com/vcharvet/geoloc-challenge

2 Methods and Data Description

2.1 Data

Our training data consists in a set of 11072334 messages received by a network of 159 base stations. Each entry describes the unique id of the message (received by several antennas) and various parameters, along which $rssi$ (Received Signal Strength Indicator), and $frequency$. In addition, we have at our disposition the coordinates of the base stations.

Each received message comes with a variety of features, some are proper to the emitting device for instance *motion*: we call them $client_features$. The others are proper to the antenna like $time_{ux}$: we call those $bs_{features}$. Consequently, to retrieve a feature vector for each message, we have to concatenate the $client_features$ with all the $bs_features$ relative to this message. Let's denote p_1 and p_2 the number of client and bs features respectively, the shape of the final feature vectors will be

$$P = p_1 + 159 \times p_2 \tag{1}$$

as there are 159 base stations. We proceed to this task using the class **Builder** from **utils.feature_builder.py**. As the computation cost for this is quite expensive, we perform it in a serialized way to reduce the execution time. Parsing the whole database with this class is usually done in less than a hour.

2.2 Methods

Even though we have a lot of features for each message, $P > 100$ when using all features with parsing technique explained in previous section, they are not really consistent for the regression task on predicting latitude and longitude: a first submission return a criterion greater than 14km ! In order to overcome this issue, we decided to estimate the distances from device to antennas, and input them into each feature vector. We tried 4 different methods for this purpose. Three of them are based on parametric estimation of the $rssi$ using ordinary least squares, and the other based on a fully connected neural network.

Basic rssi

Starting with a simple model taken from [6] section IV, we write rssi as

$$rssi = A - 10nlog(d) \tag{2}$$

where $rssi$ is the received signal strength at base station (dB), A and n are the estimated parameters and d is the distance (meters) between the device and the base station. d is calculated with geopy python package using coordinates of devices and base stations.

That means for each base station, we minimize the following objective function:

$$f_i(n, A) = \frac{1}{n_i} \sum_{j=1}^{n_i} (rssi_{ij} - A + 10\ n\ log(d_{ij}))^2 \tag{3}$$

where $i = 1 \dots 159$ indicated the base station, n_i the number of messages received by the bs, d_ij the distance between base station i and message j

Augmented rssi

To improve the previous model we use equation (3) from [2]

$$rssi = rssi_0 - 10nlog(\frac{d}{d_0}) + err \tag{4}$$

For the estimation, we will regroup $rssi_0$ and err in a unique term A and using the same notation as before, we write the objective function as:

$$f_i(n, d_0, A) = \frac{1}{n_i} \sum_{j=1}^{n_i} (rssi_{ij} - A + 10\ n\ log(\frac{d_{ij}}{d_0}))^2 \tag{5}$$

Frequency-based rssi

$$rssi = rssi_0 + 10\ n\ log(d \times f) - 30n + C \tag{6}$$

C is a propagation constant we also try to estimate even though a common used value is 32.44, f is the frequency of the received signal (MHz) This yield the following objective:

$$f_i(n, rssi_0, C) = \frac{1}{n_i} \sum_{j=1}^{n_i} (rssi_{ij} - rssi_0 + 10n\ log(d \times f) - 30n + C)^2 \tag{7}$$

We have used the module **scipy.optimize.minimize** for the estimation of parameters of the objective function described above. To overcome convergence to wrong solutions, we have constrained the value of $rssi_0$ to be greater than $max(rssi_{ij})_{j=1...n_i}$ for physical consistence (the value of the sent signal can't be less than the received one). The scripts for these estimation are available at **mains/rssi_params.py** .

Once the estimation of the parameters is done, we can estimate the distance of a new device using equations (2), (4) and (6).

Neural-based Estimation

As we have struggled finding good estimations of distances with parametric models, we tried estimated the distance with a multi-layer perceptron regressor [3]. Theoretically, such an estimator can approximate any real-valued function. Finding inspiration with parametric models described above, we tried to estimate the function f such that

$$f(lat_{bs}, lng_{bs}, freq, rssi) = d$$

That model consists in training a MLP regressor with the target d and predictors lat_{bs}, lng_{bs}, $freq$ and $rssi$ We have done this part with keras and tried different layer configurations and regularization parameters.

Infering devices coordinates

This part is the least complicated provided we have found the most adequate features in the previous tasks. However, as we find ourselves with a great number of features (1), we need to select the bet among them. As we are working in a two-target regression framework, many of the statistical tests are not applicable. Even though there exist ways to work around this, we chose simpler methods.

As our feature matrix is sparse, we put a mask over the columns to select only the ones with a number of non-zero values greater than a treshold. The latter has been tuned by cross-validation.

We also assumed the regression model would struggle training on outlier values ie targets for which the latitude or longitude is "far" from the mean: we removed the training values where $x \notin [\bar{x} \pm k \times \sigma_x]$ where x is longitude or latitude, $\bar{x}$ the mean, σ_x the standard deviation and k is a tuned hyper-parameter.

The best regressors we found were random forests and xgboost. The final predictions were made with those, with parameters tuned by bayesian optimization with hyperopt python package [1].

3 Results

Distance Estimation

Among all the techniques we used for estimating the distance using available parameters, the best was the one using MLP regression. We assume this is due to the fact that the formulas given in section 2.2 for rssi were not consistent for the wide range of distance in our dataset. They would have probably worked well for small distances only. Hence, that is the MLP estimation of the distance we used for the final predictions.

3.1 Coordinates prediction

We present our results in the table 3.1.

- *algorithms*, Random Forest (RF) or XGBoost
- $n_{features}$, number of features, we keep features that have a number of non-zeros greater than a treshold
- *mask*, range of the targets
- *CV loss*, cross validated score on 5 folds.

Algorithm	$n_{feature}$	mask	CV loss (k=5)
RF	659	None	4.9
RF	659	$\bar{x} \pm \sigma$	4.8
RF	209	$\bar{x} \pm \sigma$	6.7
RF	450	$\bar{x} \pm \sigma$	4.8
XGBoost	659	$\bar{x} \pm \sigma$	**4.6**
XGBoost	450	$\bar{x} \pm \sigma$	**5.6**

Out best regressor was xgboost with $max_{depth} = 30$ and 100 estimators. It gave a test score of 6.63 km.

4 Conclusion

Thanks to our efficient feature engineering and selection methods, we achieved a final prediction among

the best of the challenge. However, the criterion is still quite large (6.63 km).

With more time, we could have managed to improve the prediction. In particular, we think we could have obtained better estimation of the device-antenna distances with better tuning of our neural network or $rssi$ formulas better suited to the range of distances.

References

[1] Distributed asynchronous hyperparameter optimization in python http://hyperopt.github.io/hyperopt.

[2] Masoto Chiputa and Li Xiangyang. Real time wi-fi indoor positioning system based on rssi measurements: A distributed load approach with the fusion of three positioning algorithms. *Wireless Personal Communications*, 99(1):67–83, Mar 2018.

[3] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, Dec 1989.

[4] Jie He, Kaveh Pahlavan, Shen Li, and Qin Wang. A testbed for evaluation of the effects of multipath on performance of toa-based indoor geolocation. *IEEE transactions on instrumentation and measurement*, 62(8):2237–2247, 2013.

[5] Ana-Maria Roxin, Jaafar Gaber, Maxime Wack, and Ahmed Nait Sidi Moh. Survey of Wireless Geolocation Techniques. In *IEEE Globecom Workshops*, page 9 Pages, Washington, DC, United States, November 2007.

[6] M Srbinovska, V Dimcev, C Gavrovski, and Z Kokolanski. Localization techniques in wireless sensor networks using measurement of received signal strength indicator. *Electronics*, 15(1):67–71, 2011.