Neighbor Oblivious Learning (NObLe) for Device Localization and Tracking

Zichang Liu
Department of Computer Science
Rice University
Houston, USA
zichangliu@rice.edu

Li Chou
Department of Computer Science
Rice University
Houston, USA
lchou@rice.edu

Anshumali Shrivastava
Department of Computer Science
Rice University
Houston, USA
anshumali@rice.edu

***Abstract*—On-device localization and tracking are increasingly crucial for various applications. Machine learning (ML) techniques are widely adopted along with the rapidly growing amount of data. However, during training, almost none of ML techniques incorporate the known structural information such as floor plan, which can be especially useful in indoor or other structured environments. The problem is incredibly hard because the structural properties are not explicitly available, making most structural learning approaches inapplicable. We study our method through the intuitions from manifold learning. Whereas existing manifold methods utilizes neighborhood information such as Euclidean distances, we quantize the output space to measure closeness on the structure. We propose Neighbor Oblivious Learning (NObLe) and demonstrate our approach's effectiveness on two applications, Wi-Fi-based fingerprint localization and inertial measurement unit(IMU) based device tracking. We show that NObLe gives significant improvement over state-of-art prediction accuracy.**

I. Introduction

Past years have witnessed a critical need for accurate location information. For example, location intelligence is essential during public health emergencies, such as the COVID-19 pandemic, where governments need to identify infection sources and spread patterns. Traditional localization systems rely on global positioning system (GPS) signals; however, GPS can be inaccurate under specific scenarios such as indoor environments and city skyscrapers. Moreover, GPS is notorious for battery drainage because of slow and demanding communication requirements [1]. Therefore, GPS alternatives with higher precision and lower energy consumption are urged by industry. Existing network infrastructure such as Wi-Fi (IEEE 802.11) is adopted for localization [2] [3] to avoid expensive infrastructure deployment. Low-cost inertial measurement sensors (IMU), which are widely embedded in modern mobile devices, have also emerged as a popular solution [4] [5] for device tracking. An informative and robust estimation of position based on these noisy inputs would further improve localization precision. Machine learning techniques are logical choices, and popular algorithms such as k-nearest neighbors and random forest have been used [6] for accurate localization. Recently, attempts are also made to utilize deep neural network (DNN) [7] [8] [9] for its promising performance. These approaches either formulate localization as minimizing distance errors or use deep learning as denoising techniques for robust signal features.

All the methods mentioned above fail to utilize common knowledge: space is highly structured. Modern city planning defines all roads based on specific rules. Indoor space is structured by its floor plan. Consider Fig 1 based on the largest publicly available indoor Wi-Fi localization dataset UJIIndoorLoc [10]. Space structure is clear from the satellite view, and offline sampling locations exhibit the same structure. Fig 2(a) shows the outputs from a DNN that is trained using mean squared error to map Wi-Fi signals to location coordinates. Not surprisingly, this regression model predicts locations outside of buildings as it entirely ignore the output space structure. It was observed in [5] [11] that projecting the predicted outputs to the closest positions on the map would increase precision. Our experiment shows that forcing the prediction to lie on the map only gives marginal improvements.

Fig. 1: Left figure is the screenshot of aerial satellite view of the buildings (source: Google Map). Right figure shows the ground truth coordinates from offline collected data.

Given that the space contains rich structures, our problem can be regarded as learning a regression model in which the input and output lie on an unknown manifold. When explicit manifold distances are unavailable, Euclidean distance near neighbors is used as an approximation for the closeness among points on the actual manifold. However, the Euclidean distances may not be a good criterion for localization services as input signals are extremely noisy.

Inspired by approaches in manifold learning, we propose to ignore small changes in the Euclidean distance and focus on the relative closeness of reconstructed embedding. We propose **N***eighbor* **Ob***livious* **Le***arning* (NObLe) for structure-aware localization. We demonstrate our techniques' applicability on two independent applications: Wi-Fi based indoor localization and IMU-based outdoor device positioning. Our evaluations on both applications show that NObLe gives significant accuracy improvements. To illustrate that our system can be deployed

978-3-9819263-5-4/DATE21/©2021 EDAA

on energy constraints mobile devices, we measure the energy usage for both applications on Nvidia Jetson TX2. A longer version of this paper with more details can be found at http://arxiv.org/abs/2011.14954

II. Background and Related Work

Manifold Learning: The objective of manifold learning is to find a low-dimensional representation describing given high-dimensional data from an input space $\mathcal{X} \subset \mathbb{R}^d$, that is, learn a mapping $\psi : x \to z \in \mathbb{R}^s$ such that $s \ll d$ while, loosely stated, preserving (structural) properties (e.g., interpoint distances) of the original space. Two popular manifold learning methods are locally linear embedding (LLE) [12] and isometric mapping (Isomap) [13]. These algorithms follow a template comprised of three steps: (1) construct a neighborhood graph, which involves (expensive) nearest neighbor search; (2) construct a (positive semi-definite) kernel, which is specified as shortest path distances for Isomap, and weights (or coefficients) from solving a system of linear equations for LLE; and (3) perform partial Eigenvalue decomposition.

Wi-Fi Localization: Wi-Fi localization consists of two phases. In the offline phase, signal features such as the received signal strength indicator (RSSI) are sampled at selected locations and processed to build a database of locations with their corresponding signal values. In the online phase, observed signals are matched with the most similar points from the database to determine current locations.

IMU Localization: IMUs are extremely noisy, making it impossible to use only through physical principles and numerical integration. IMU localization usually keeps updating previous positions, which makes it subject to error accumulation. A line of work utilizes maps to hand-design heuristic rules to correct localization error by ruling out illegal movements.

ML in Localization: ML approaches formulate localization as a regression problem that predicts continuously coordinate variables given signal strength. ML is also used as denoising techniques to extract core signal features. WiDeep [8] utilize one auto-encoder (AE) for every WAP, making it hard to scale. DeepFi [7] also uses DNNs, but also ignore structure information. CNNLoc [9] utilizes a complex architecture including stacked AEs and convolution neural networks to achieve a mean error of 11.78m on UJIIndoorLoc. [5] used nearest neighbors and random forest to predict the travel distance based on IMU.

III. Proposed System Design

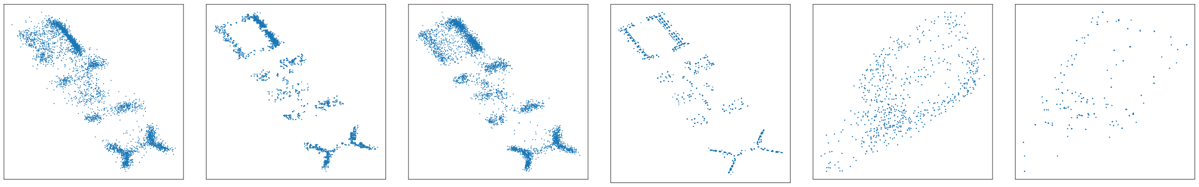
Given the structural nature of localization space, we approach the problem considering that the input and output space lies in a manifold space. Manifold-based algorithms utilize local Euclidean distances to approximate manifold neighborhood structure. However, the input features for localization are extremely noisy. When a person is walking, the accelerometer and gyroscope sensors are prone to noises due to spurious movements. Similarly, Wi-Fi signals can be noisy because of moving crowds or room set-ups. Thus, small changes in such noisy input signals are not reliable information about the structure and direct adopting traditional manifold learning approaches is not appropriate. To combat the noise, we ignore small Euclidean differences and propose Neighbor Oblivious Learning (NObLe). We propose to quantize the continuous output space into a set of grid-like neighborhood areas, and all data points within the same grid are considered belonging to the same class. It is widely accepted that the penultimate layer of the deep neural network classifier model can be regarded as learned embedding [14]. We use DNN and optimize it with cross-entropy loss to maximize the embedding distance between different classes while oblivious to embedding distance within the same class.

Space Quantization: Consider a space S for localization, we collect data samples of the form $(\vec{s}, (x, y))$, where $\vec{s}$ is a vector for signal features, and (x, y) denotes longitude and latitude coordinates. We propose to quantize (x, y) to transform continuous position coordinates into neighborhood area classes. Each data sample becomes $(\vec{s}, c, (x, y))$, where c is a neighborhood area classes ID. Specifically, we divide S into non-overlapping square grids with a side length of τ. In practice, we set τ to be less than 0.2m. Then, we assign each grid neighborhoods a class ID c and discard all classes without any data points. Thus, instead of using position coordinates as training labels, NObLe uses neighborhood class as ground truth. During inference, NObLe uses the predicted class to look up its neighborhood class's central coordinates as the prediction result. Noted that our evaluation measure is position error (root mean square error). Our approach approximates the ground truth closeness between data points in the output space without relying on Euclidean distance in the input space. Moreover, with a thorough training data sampling process over space S, our method eliminates inaccessible areas from the output space. In Fig 1, the middle area of top left buildings will not translate to any neighborhood classes as no data resides in that area.

Multi-label Classification: Our space quantization solves the manifold regression problem with a classification model. However, we have introduced one hurdle. The classification problem is likely to suffer from class data sparsity because our grids are fine-grained, and each grid is likely to contain very few data samples. We assign data samples with multiple classes, the ones that are adjacent to the real class. Moreover, we divide space S into grid neighborhood of different length, τ and l where $\tau < l$. Each data sample becomes $(\vec{s}, c, r(x, y))$ where c denotes for the neighborhood classes ID of size τ and r denotes neighborhood classes ID of size l. This formulation gives various granularity levels of the manifold.

Why DNN Classification is Equivalent to Manifold Learning: We will make the connection between manifold learning and our approach mathematically. We introduce multidimensional scaling (MDS), a popular manifold learning algorithm, which has the objective: $f(Z, \mathcal{X}) = \sum_{i=1}^{n} \sum_{j=1+i}^{n} (||z_i - z_j|| - ||x_i - x_j||)^2$ for n points. MDS tries to learn embedding Z such that close neighbors are encouraged to stay close in the reconstructed space and vise versa.

In NObLe, we use binary cross-entropy loss function for multi-label classification, defined as $J(h_c, \hat{h}_c) = \sum_{i=1}^{n} \sum_{c=1}^{k} -h_c \log(\hat{h}_c) - (1 - h_c) \log(1 - \hat{h}_c)$, where k is the number of classes, n is the number of training data, $h_c \in \{0, 1\}$ indicates the right class when $h_c = 1$. $\hat{h}_c$ is the Sigmoid



(a) Deep Regression **(b)** Deep Regression Projection **(c)** Isomap Regression **(d)** NObLe **(e)** Deep Regression **(f)** NObLe

Fig. 2: (a)(b)(c)(d) are plots of predicted coordinates for Wi-Fi localization. (e) and (f) are plots of predicted coordinates for IMU device tracking. Methods are labeled below

function: $\hat{h}_c = (1 + \exp(-w_c^\top z_i))^{-1}$ where w_c denotes the weight vector for class c at the last layer, and z_i denotes for the output from the second last layer for input x_i. We focus our analysis on the last layer because the second last layer output can be interpreted as learned embedding for input features. From a manifold learning perspective, embedding from the last layer can be interpreted as reconstructed embedding. For simplicity, suppose w and z are normalized, we can rewrite $\hat{h}$ from an inner product to the Euclidean distance form as $\hat{h}_c = (1 + \exp(\frac{1}{2}||w_c - z_i||^2 - 1))^{-1}$.

For a given c, minimizing the cross entropy loss will result in a setting such that $||w_c - z_i||$ for the true class is minimized (cf. false class is maximized). Consider z_i as embedding given input x_i, z_j as embedding given input x_j. If x_i, x_j are near neighbors, then by our formulation, x_i, x_j share same class label. Thus, the following holds for two embedding z_i and z_j, $||w_c - z_i||^2 \leq \lambda$ and $||w_c - z_j||^2 \leq \lambda$, where λ is a small constant. And by triangle inequality, we have $||z_i - z_j||^2 \leq 2\lambda$. As we can see, z_i, z_j is expected to be close, which resembles the objective function of MDS without considering the distance in the input space between x_i and x_j.

IV. Application I: Wi-Fi Localization

In this section, we present NObLe for Wi-Fi localization following the standard Wi-Fi fingerprint setting. Each offline sampled data can be represented as $(\vec{s}, b, f, (x, y))$. $\vec{s} = (s_1, s_2, \ldots, s_W)$, where s_i denotes the RSS of i-th WAP, b denotes building ID, f denotes floor ID. We perform output space quantization and convert each sample to $(\vec{s}, b, f, c, r, (x, y))$. Our model takes $\vec{s}$ as inputs, and predict (b, f, c, r). We use c to look up the corresponding central coordinates, and output (x_c, y_c) as position and calculate position error accordingly.We use a two hidden layer feed-forward neural network with hidden size 128. NObLe naturally include floor/building classification, a standard task for localization. Current approaches utilize separate and independent models for position prediction and building/floor classification, creating extra overhead in real-world deployments.

Evaluation: We use two representative indoor Wi-Fi localization datasets: UJIIndoorLoc [10], the largest open-access dataset for indoor Wi-Fi localization and IPIN2016 [15]. We calculate position error following the standard procedure as the Euclidean distance between predicted and true coordinates.

On UJIIndoorLoc, the best mean error distance on the indoor localization ranking at IndoorLocPlatform website [15] is 6.2 m, and the median is 4.63m. [9] reports a mean position error of 11.78 m, a building hit rate around 99%, and a floor hit rate around 94%. [16] reports a mean position error of 9.29m, a building hit rate around 99%, and a floor hit rate around 91%. As we can see from Table I, NObLe achieves significantly smaller position error distances and at least comparable building and floor hit rate.

TABLE I: NObLe performance on UJIIndoorLoc.

	Classification Accuracy (%)
Building	99.74
Floor	94.25
Quantize Class	61.63
	Position Error Distances (m)
Mean	4.45
Median	0.23

To evaluate the performance from the perspective of structure awareness, we implement three comparison models. Deep Regression uses mean square error as a loss function and directly predicts coordinates. Deep Regression Projection projects the predicted coordinates from Deep Regression to the nearest position on the map when the predictions do not lie on the map. Given signal embedding learned by Isomap and LLE as input, LLE/ISOMAP Deep Regression directly predicts coordinates. The best performance is achieved with embedding dimension at 400 for both Isomap and LLE.

TABLE II: Position error distance (m) on UJIIndoorLoc.

Model	Mean	Median
Deep Regression	10.17	7.84
Regression Projection	9.76	7.16
Isomap Deep Regression	11.01	7.56
LLE Deep Regression	10.05	7.43

Fig. 2(a), 2(b), 2(b), and 2(d), are plots of predicted coordinates on the UJIIndoorLoc dataset. NObLe outputs the most structured prediction. A considerable number of the Deep Regression predictions lie in the middle are of the top left building, which is not part of the buildings. Isomap Deep Regression is visually more structured than Deep Regression, which is expected because Isomap Embedding is reconstructed to approximate output structure. Deep Regression Projection resembles the building structure because it corrects prediction based on human-crafted maps.

On IPIN2016, NObLe achieves an average error distance of 1.13m and a median average error distance of 0.046m, while the Deep Regression gives an average error distance of 3.83m.

The best mean error distance on the indoor localization ranking at IndoorLocPlatform website [15] is 3.71m.

Using UJIIndoorLoc, the average running energy for each inference is 0.00518J, and the average latency is 2 milliseconds.

V. Application II: Device Tracking Using IMUs

We present NObLe for device tracking using IMU signals. A user travels along a specific path, and a sequence of IMU data corresponding to this travel path is recorded. Given the signals, we want to predict the user location at the end of the path. Without available public datasets, we collect labeled data in an outdoor space for device tracking.

The input consists of an initial location coordinates h_{start} and a sequence of IMU signals $G = g_1, g_2, \ldots$, where $g_i \in \mathbb{R}^{d\times n}$. d is the dimension of each inertial sensor readings, and n is the number of sensors. We perform output space quantization at $\tau = 0.4$m and assign neighborhood classes c for path ending location. Our model calculates ending position in longitude and latitude based on predicted neighborhood class $\hat{c}$. Our system includes three main parts: projection module, displacement module, and location module. The projection module takes g_i and outputs an embedding in a lower dimension using the same trainable projection weights. All projection embeddings are concatenated and pass into the displacement module, a two-layer feed-forward neural network that predicts the displacement vector of a user's travel path. This module is not environment-specific, and a trained module can be plugged into other models for location tracking in other environments. Taken the outputs from the displacement module and one-hot encoded starting location, the location module outputs location class at the end of the travel path.

Data Collection: We collect data from two independent walks in an area of 160m by 60m on the campus of Rice University. The sampling frequency is 50Hz, and the total walking time is 1.3 hours. There are 177 reference locations with GPS coordinates. Between each reference point, there are 768 readings for each inertial sensor on a single axis. We record 3-axis gyroscope, 3-axis accelerometer, and timestamps. We construct 6857 paths in total, and 4389 are used for training.

TABLE III: Position error distance (m) for IMU tracking.

	Mean	Median
Deep Regression Model	10.41	10.05
[5]	4.3	N/A
NObLe	2.52	0.4

Evaluation: We implemented Deep Regression similar to Wi-Fi localization experiment and also use [5] for comparison. Fig.2(e) and 2(f) shows that NObLe performs better in capturing the structural information. The predicted location points are less scattered around comparing to Deep Regression and resemble the space structure more closely. Position errors are summarized in Table III. NObLe gives the smallest location error. [5] iterative corrects prediction location at all turnings on the path. LocMe [11] reports a median of 1.1m position error on test-bed size of 70m by 100m by constantly correcting at elevators and walls. We could not test their method on our dataset as they did not open source their code. Both [5] and LocMe [11] systems human effort to transfer map knowledge into heuristic rules.

For a testing path for around 8 seconds, NObLe consumed around 0.08599J for inference with a 5 milliseconds latency. The inertial sensor's energy cost is 0.1356J for 8 seconds. The total energy consumption is approximately 0.22159J, which is $27\times$ less than the GPS energy requirement 5.925J based on [5].

VI. Acknowledgement

This work was supported by National Science Foundation IIS-1652131, BIGDATA-1838177, AFOSRYIP FA9550-18-1-0152, ONR DURIP Grant, and the ONR BRC grant on Randomized Numerical Linear Algebra.

References

[1] T. O. Oshin, S. Poslad, and A. Ma, "Improving the Energy-Efficiency of GPS Based Location Sensing Smartphone Applications", 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, 2012, 99.1698-1705

[2] P. Bahl and V. N. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064), Tel Aviv, Israel, 2000, pp. 775-784 vol.2

[3] A. M. Ladd, K. E. Bekris, A. P. Rudys, D. S. Wallach and L. E. Kavraki, "On the feasibility of using wireless ethernet for indoor localization," in IEEE Transactions on Robotics and Automation, vol. 20, no. 3, pp. 555-559, June 2004

[4] Q. Yuan and I. Chen, "Localization and velocity tracking of human via 3 IMU sensors," Sensors and Actuators A: Physical, Volume 212, 2014, Pages 25-33, ISSN 0924-4247

[5] E. J. J. Gonzalez et al., "Location detection for navigation using IMUs with a map through coarse-grained machine learning," Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017, Lausanne, 2017, pp. 500-505

[6] M. A. Youssef, A. Agrawala and A. Udaya Shankar, "WLAN location determination via clustering and probability distributions," Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003. (PerCom 2003)., Fort Worth, TX, 2003

[7] X. Wang, L. Gao, S. Mao and S. Pandey, "CSI-Based Fingerprinting for Indoor Localization: A Deep Learning Approach," in IEEE Transactions on Vehicular Technology, vol. 66, no. 1, pp. 763-776, Jan. 2017

[8] M. Abbas, M. Elhamshary, H. Rizk, M. Torki and M. Youssef, "WiDeep: WiFi-based Accurate and Robust Indoor Localization System using Deep Learning," 2019 IEEE International Conference on Pervasive Computing and Communications (PerCom, Kyoto, Japan, 2019, pp. 1-10

[9] X. Song et al., "A Novel Convolutional Neural Network Based Indoor Localization Framework With WiFi Fingerprinting," in IEEE Access, vol. 7, pp. 110698-110709, 2019

[10] J. Torres-Sospedra et al., "UJIIndoorLoc: A New Multi-Building and Multi-Floor Database for WLAN Fingerprint-Based Indoor Localization Problems," 2014 Int Conf Indoor Position Indoor Navigation Ipin, 2014

[11] Xinye Lin and Xiao-Wen Chang and Xue Liu, "LocMe: Human Locomotion and Map Exploitation Based Indoor Localization," 2017 IEEE International Conference on Pervasive Computing and Communications (PerCom), pp. 131–140, 2017

[12] S. T. Roweis and L. K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," Science, vol. 290, no. 5500, pp. 2323–2326, 2000

[13] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," Science, vol. 290, no. 5500, pp. 2319–2323, 2000

[14] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives.", International Conference on Learning Representations, 2014

[15] J. Torres-Sospedra et al., IPIN2016 Tutorial, http://indoorlocplatform.uji.es/databases/get/2/

[16] K. S. Kim, S. Lee, and K. Huang, "A scalable deep neural network architecture for multi-building and multi-floor indoor localization based on Wi-Fi fingerprinting," Big Data Anal, vol. 3, no. 1, p. 4, 2018