2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining

LocationTrails: A Federated Approach to Learning Location Embeddings

Saket Gurukar[1], Srinivasan Parthasarathy[1], Rajiv Ramnath[1], Catherine Calder[2] and Sobhan Moosavi[1]
gurukar.1@osu.edu, srini@cse.ohio-state.edu, ramnath.6@osu.edu, calder@austin.utexas.edu, smoosavi@osu.edu
[1] Department of Computer Science and Engineering, The Ohio State University
[2] Department of Statistics and Data Sciences, University of Texas at Austin

***Abstract*—Learning a vector representation of locations that reflect human mobility patterns is useful for various tasks, including location recommendation, city planning, urban analysis, and even understanding the neighborhood effects on individuals' health and well-being. Existing approaches that model and learn such representations either do not scale or require significant resources to scale. They often need the entire data to be loaded in memory along with the intermediate data representation (typically a co-location graph) and are usually not feasible to execute on low-resource embedding systems such as edge devices. The research question we seek to address in this article is, can one develop efficient federated learning models for location representation learning such that the training and the subsequent updates of the model can occur on edge devices? We present a simple yet novel model called LocationTrails for learning efficient location embeddings to address this question. We show that our proposed model can be trained under the federated learning paradigm and can, therefore, ensure that the model can be trained in a distributed fashion without centralizing locations visited by all users, thereby mitigating some risks to privacy. We evaluate the performance of LocationTrails on five real-world human mobility datasets drawn from two use cases (four of them from driving trajectory data obtained from a national insurance agency; and one of them from a unique study of adolescent mobility patterns in an urban setting). We compare our proposed LocationTrails model against the strong baselines from the network representation learning field. We show the efficacy of LocationTrails in terms of better embedding quality generation, memory consumption, and execution time. To the best of our knowledge, the federated LocationTrails model is the first model that can generate efficient location embeddings without requiring the complete data to be loaded on a central server.**

***Index Terms*—Human Mobility Analysis, Federated Learning, Representation Learning**

I. Introduction

The widespread availability of GPS-enabled devices such as mobile phones and other edge devices has made it much easier to capture and store human mobility data. Leveraging such data sources is helpful in urban analysis [1], city planning [2] and even understanding the neighborhood effects on individuals' health and safety [3]. Importantly, learning a vector representation of locations that reflect such mobility patterns can be incredibly useful in improving services such as location recommendation [4], classification of Points Of Interest (POIs) [5] and venue mapping [6].

A natural model to represent such human mobility data is to lever a co-location network [7]. A co-location network is a bipartite (two-mode) network where one partition comprises users, and the other partition comprises locations. If a user visits a location, it is represented by an edge. Each edge can optionally include attributes (time of visit, number of visits). Each node can also optionally include attributes (GPS coordinates, location labels, user attributes). Given this abstraction, one can leverage techniques proposed in graph representation learning to understand and query human movement dynamics through co-location graphs.

The main challenge in adopting such ideas for our purpose is that such solutions are often not feasible for low-resource embedded systems or edge devices that need to accommodate both training and inference [8]. First, scaling such graph representation learning methods to the limited computational and memory budget of such devices is challenging [9]. Second, most of the methods surveyed rely on a global (or centralized) picture of the graph model, which in turn can exacerbate privacy[1] concerns with eventual deployment. We note that AI at the edge is a shift in the computing paradigm from cloud computing with advantages such as faster real-time decisions, offline availability, reduced bandwidth consumption, and also potentially mitigates some privacy pitfalls as the raw data from consumers is not shared to a centralized cloud server. Such ideas have found initial deployment in human activity recognition [11], medical diagnostics [12] and mobile keyboards [13], among others.

In this paper, we describe an approach – LocationTrails – that seeks to address these challenges. Key to our approach is to build localized co-location patterns specific to an individual user. The constructed sequence pattern can be abstractly viewed as a path on the co-location network but importantly

[1]Many users would not prefer sharing the location visit data with the organizations. Moreover, in Europe, the EDPB guidelines [10] mandates that users consent to access data from their smartphones or **from their vehicles**. As a result, it becomes critical to develop privacy-preserving mobility models that can perform human mobility analysis and also satisfy the EDPB guidelines.

ASONAM '21, August 27-30, 2021, The Hague, The Netherlands

ACM ISBN 978-1-4503-9128-3/21/11
10.1145/3487351.3490964

limited to only those parts of a network an individual user actually visits. In other words, the global network is not explicitly leveraged by LocationTrails. However, as described later, LocationTrails implicitly leverages the common locations visited by users. Given a path on the co-location network, one can then extract pairs of target and context tokens from these sequence patterns, computed using a chosen context window. These pairs of target and context tokens are then fed to the skip-gram model [14] for learning the location embeddings. We note that this step can be executed in two different settings: 1. On a single server with optimization performed through batch gradient descent; and 2. In a distributed setting by adapting the idea of federated learning [15] for training the model. Federated learning approach trains a model across multiple decentralized edge devices without exchanging their raw data samples. In other words, at each step, training is performed on a set or subset of edge devices (according to a specific algorithmic protocol) and the representation of the model is the only information that is shared with the server. This methodology allows one to tune the tradeoff between privacy and performance depending on the number of decentralized clients used to model the optimization.

Note that the existing graph embedding methods require the complete graph to either perform matrix factorization or random walks and hence cannot be easily trained within a federated learning setup. In contrast, our proposed method offloads the location embedding training (and inference) computation from the central server to edge devices, which is vital for applications whose services are availed by millions of users daily. The learned location embeddings can then be treated as a distributed representation of the location for downstream applications such as location recommendation, next location prediction, and types of locations a user may visit during the course of a day. Since the training instance of LocationTrails is constructed from locations visited by a user on a single day, the model can be updated with the new data by treating the new data as a new batch for training. However, the existing graph embedding methods need to be trained from scratch, as with new data, the graph structure would change.

We evaluate the performance of LocationTrails along the axes of quality (of generated representations), resource requirements (computational and memory). We also assess the impact of federated learning, especially on quality (for example, it has been shown in other studies that there is a qualitative cost associated with federated training – i.e., improved mitigation of privacy concerns comes at a quality cost). Our evaluation focuses on five datasets – four of which model human mobility patterns from driving trajectories in four major US cities; and one of which models human mobility of adolescents within an urban (inner-city) setting. For the former, provided by a major insurance company, such location representation models can be useful for characterizing driving style and driving context [16], [17]. For the latter, location representation can provide cues on the types of locations each user is likely to encounter during their daily routine and whether such locations may inform on exposures to violence, risky behavior, or collective efficacy. In both use-case settings, node classification is an important cue for qualitative performance, so we compare the performance of LocationTrails and the competitive strawman on this specific task.

Our experiments on the real-world human mobility data collected from a major insurance company and the adolescent mobility study from a major US city demonstrate the efficacy of our proposed method in terms of better embedding quality, less memory consumption, and faster execution than the popular graph embedding methods. The contributions of our work can be summarized as

- We propose a simple yet novel LocationTrails model for learning efficient location embeddings.
- Our proposed LocationTrails model is amenable to be trained via Federated Learning and, therefore, can preserve users' privacy and offload the computation burden from server to edge devices.
- We perform experiments on five real-world human-mobility datasets collected from distinct application domains and show the efficacy of the proposed method on both single server and federated system.

II. Related Work

Location embeddings models: Location2vec [18] learns location representations by collecting the Geo-tagged tweets which fall within 10 meters of the selected location. The authors then lever the skip-gram model [14] and treat the location token as the target word and treat the words in the Geo-tagged tweets as context words. A key differentiator from our effort, is that they learn the representation from textual data while we leverage (co-)location visit data collected by GPS-enabled sensors. LBSN2vec [19] studies user mobility and social relationships in Location-Based Social Networks. The authors collect user check-ins and users' social network and then propose a hypergraph-based random walk approach to learn location embeddings. Note that the social network of users is not always available. Yan et al. [5] learn the Point of Interest (POI) embeddings by proposing a novel method of training corpus generation based on augmented spatial contexts for word2vec model [14]. Few methods [20]–[22] focus on the problem of Point-of-Interest (POI) classification. However, these methods have been proposed for check-in data and require additional information such as text content of POI [22], or their source code is not publicly available [20], or hard to reproduce due to lack of documentation and sample data [21]. CATAPE [23] focuses on the problem of top-k POI recommendation and propose a two-tower recommendation model on the user embedding and POI embedding. The POI embedding is learned by employing skip-gram on the user's POI check-ins sequence and POI categories sequence. As we shall shortly see, both the problem setup and model architecture are distinct from our approach. Moreover, the user's privacy concerns and distributed training aspects are not considered in CATAPE [23].

Word embeddings: Mikolov et al. [14] proposed a skip-gram model that attempts to maximize the probability of con-

text words given the target word. Pennington et al. [24] levers the global word-word co-occurrence matrix and achieves a good vector space structure for tasks such as word-analogy. Recently, transformer [25] based models such as BERT [26] have shown tremendous promise on multiple natural language processing tasks such as natural language inference and entailment. However, the transformer-based models require a massive amount of data, which is not always available.

Graph Representation Learning: Recently, there has been surge of works in graph representation learning [27]–[30]. Deepwalk [27] learns node embeddings in the graph by first performing multiple unbiased random walks of fixed length from all the nodes in the graph and then employs the skip-gram model to learn the node embedding. Node2vec [28] proposed a biased random walk algorithm that alternates between Breadth-First Search and Depth-First Search to achieve a microscopic or macroscopic view of the graph, respectively. Line [31] learns node embeddings by maximizing first-order and second-order proximity between the nodes. Qiu et al. [29] showed that the network embedding methods such as Deepwalk [27], Node2vec [28], and Line [31] are implicitly performing matrix factorization on closed-form graph Laplacian. Deep Graph Infomax [30] learns node embedding by maximizing the mutual information between the global representation of the graph and the local information of the node. BINE [32] performs a biased random walk on bipartite network where the number of random walks for each node varies based on the nodes' importance while the walk length is determined in a probabilistic way. The interested reader is encouraged to refer to surveys [33], [34].

III. Methodology

Here, we present our proposed approach LocationTrails and begin by describing our graph abstraction model.

Co-location Graph Model: Human mobility data can often be represented as through a bipartite co-location network $\mathcal{G} = (U, L, E)$ where U and L are the set of all users and locations, respectively, and E denotes the set of edges where an edge (u, l, t) is formed between user u and location l if user u visits location l at time t. The Figure 1a shows the user-location co-location network.

LocationTrails: Our proposed method consists of two stages: in the first stage, we construct a sequence based on the location visit information of the users. This sequence can be abstractly viewed as a walk on the co-location network but is limited to the locations (or a subset of locations) actually visited by the user. Let a location i be denoted as L_i, a user j be denoted as U_j, let D_l be the date l, W_e and W_d represent weekend and weekday, respectively. Additionally, let D be the set of all dates. So, a user U_j visits locations on a particular day W_e would be represented as:

$$U_j : D_l : L_{i_1},\ L_{i_2},\ L_{i_3}, L_{i_1},\ \ldots\ L_{i_n}$$

The sequence of locations is temporally ordered such that if location L_i is visited by user U_j at t_p and $L_{i'}$ is visited by user U_j at t_q where $p < q$ then the L_i would appear earlier in the sequence than $L_{i'}$. Note that the sequence can include repeats – i.e., the same location is visited more than once (e.g., home, school, etc.). Preserving such repeat visits ensures that the proximity of the eventual representation of the user node and the location node remains close.

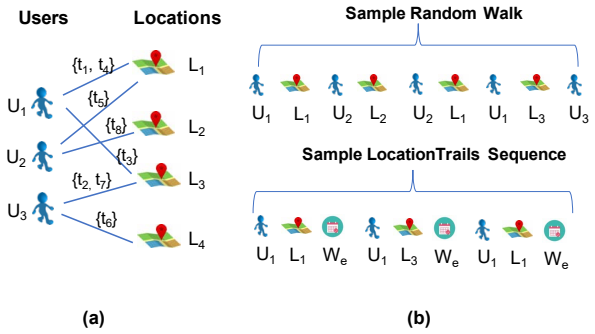


Fig. 1. The bipartite co-location network between users and locations. The edge represents users' visit to a location, t_i denotes the time of the visit and W_e denotes weekend. We also show the sample random walk from user U_1 and the LocationTrails sequence constructed for user U_1.

Now given a sequence of locations visited by a user over a given temporal resolution (e.g., a day), we construct a modified sequence where we represent each location with $(location\ id, user\ id, weekday\ /weekend\ id)$. So, from the previous example, if the day of the date D_l is weekend, then the newly constructed sequence of the previous example sequence would be represented as:

$$U_j, L_{i_1}, W_e, U_j, L_{i_2}, W_e, U_j, L_{i_3}, W_e, U_j, L_{i_1}, W_e, ..., L_{i_n}, W_e$$

Figure 1b shows a sample LocationTrails sequence constructed for user U_1 where the user visits locations in the following temporal order on a particular weekend: L1, L3, L1. Notice that the constructed sequence does not explicitly take into account the other users. While this is useful from a privacy-preserving perspective, it can be a limitation w.r.t. traditional graph representation learning methods which do take other users into account (see Figure 1). However, since LocationTrails is jointly representing users and the locations they visit – if other users also visit similar locations, the information from other users will be implicitly represented by our model.

We also point out that the user ids and weekday/weekend ids in the constructed sequence enforce the following constraints in the embedding space. The inclusion of user ids enforces that the locations visited by the same users are implicitly brought closer in embedding space, while the inclusion of weekday/weekend ids enforces that the locations visited during the same day of the week are implicitly brought closer in embedding space. Therefore, this inclusion of ids results in implicit regularization in the skip-gram objective function. We empirically show (in the ablation study in the experiments section) that the addition of this implicit regularization helps in improving the quality of embeddings.

In the second stage, we consider the modified sequence as analogous to a sentence in the Word2Vec skip-gram model

[14]. The training corpus of such skip-grams consists of a pair of target and context word and are constructed from the input sentence based on the context window size. The skip-gram model attempts to maximize the probability of the context word given the embedding of the target word. The objective function is given as:

$$minimize_{\phi} \sum_{t_j \in C(t_i)} -log\ Pr(t_j|\phi(t_i)) \tag{1}$$

where t_i can be either user, location or weekday/weekend token, and $C(t_i)$ be the context words of token t_i. Also, let $\phi(t_i)$ be the embedding of token t_i. The probability of the context word given the target word is computed with a softmax layer, a log-linear model. The softmax term is given as:

$$Pr(t_j|\phi(t_i)) = \frac{\exp\left(\phi(t_j)\phi(t_i)\right)}{\sum_{t_k \in \{L,U,W_e,W_d\}} \exp\left(\phi(t_k)\phi(t_i)\right)} \tag{2}$$

However, to reduce the computational burden of the above softmax term, we employ a standard negative sampling [14] technique for approximating the softmax and is specified as

$$log\ Pr(t_j|\phi(t_i)) = log\ \sigma(\phi(t_j).\phi(t_i)) + \sum_{p=1}^{m} \mathbb{E}_{t_k \sim P_t(t_i)}[\ log\ \sigma(-\phi(t_k).\phi(t_i))\ ] \tag{3}$$

where σ is the sigmoid function, m is the number of negative samples and $P_t(t_i)$ is the noise distribution of all the location, user and weekday and weekend tokens.

The pseudo-code of LocationTrails is shared in Algorithm 1. For each user i from the list of available users and on each available date d for which we have the location visit information of user i, we construct a sequence of locations visited by the user i on date d. Then, we replace each location with the tuple $(location\,id,\ user\,id, weekday/weekend\,id\,of\ date\ \ d)$ while maintaining the order of the locations visited. The modified sequence is treated as a sentence and a pair of target words and context words is computed. These target, context pairs are then fed to the skip-gram model described in equations 1 and equation 3.

Federated LocationTrails: We now describe the federated training procedure of LocationTrails. Federated learning [15] is a distributed machine learning approach that performs iterative model averaging through synchronous batch training. This approach is useful in unbalanced and non-IID data distributions. Note that the sensitive and private client data present on an edge device is not shared with the server but only the model updates are shared.

The training procedure of the LocationTrails model with Federated Learning is described as follows: 1) The client downloads the initial (base) LocationTrails model from the server. 2) The client then collects the mobility data of the user and trains the LocationTrails model using the training procedure specified in Algorithm 1. The user embeddings are stored locally at the client [35] and are not shared with the server. The negative samples required for Equation 3 is constrained to be either location ids or week ids. The client then updates the model locally (Equation 4). 3) Then the client sends the updated model – consisting of embeddings of all the tokens (except user embeddings) – to the server. 4) The server collects the updated models from multiple clients and then performs model averaging in a weighted fashion (Equation 5). Then steps 1-4 are repeated for n number of epochs.

Algorithm 1 LocationTrails

Input: Sequences of users' location visits along with day information. Context window w.
Output: Location embeddings.

sentences = {}
for each user i in U **do**
  for each date d in D **do**
    Collect and sort the locations by user i on date d in the visited order.
    Replace each location with the tuple $(location\,id,\ user\,id, weekday/weekend\,id\,of\ date\,d)$
    Add the sequence constructed in previous step to the list of *sentences*
  end for
end for
Construct target and context pairs from the sentence available from the list of *sentences*.
Learn the location embeddings using Eqn 1 and Eqn 3.

Let $l(\phi)$ be the loss defined in equation 3, w_t be the current global model present at the server and w_t^k be the model present at client k. Let η be the local learning rate. Then using the standard gradient descent the model w_t^k would be updated as:

$$w_{t+1}^k = w_t^k - \eta \nabla l(\phi) \tag{4}$$

Let C_t be the set of clients which trained the LocationTrails model w_t and then sent the updated model w_{t+1}^k to the server, n_k be the training instances at client k and n be the total number of training instances. The current global model w_t present at the server is updated as:

$$w_{t+1} = \sum_{k=1}^{|C_t|} \frac{n_k}{n} w_{t+1}^k \tag{5}$$

The above approach ensures that raw data from the clients is never directly shared with the central server. However, this approach is not perfect as the client could still lose some privacy if the updates to the central server only impact a fraction of the locations being modeled. In this scenario, one could reverse engineer information about locations visited. The privacy-preserving nature of this scheme could potentially be further improved via differential privacy [36] coupled with secure aggregation protocols [37]. We do not explore this thread or other forms of adversarial attacks [38], [39] on federated location trails at this point but this is a direction we would like to explore in the future.

IV. Experiments

A. Datasets

The first use-case consists of driving trajectory data obtained from a national insurance agency *Nationwide Insurance*; while the second use-case consists of adolescent mobility patterns in an urban setting obtained from the *Adolescent Health and Development in Context (AHDC) Study*. Table I shows the statistics of the five datasets. We consider a sequence of locations visited by an individual on a particular day as a trail. The statistics Mode and Mean are computed on the distribution of the length of all the trails.

1) Nationwide Trajectories: The "*Nationwide* Trajectories" is a real-world dataset of telematics information provided by *Nationwide*. This dataset provides comprehensive data collected by designated devices on each connected vehicle. The data used in this study were collected from Atlanta (GA), Columbus (OH), Pittsburgh (PA), and Philadelphia (PA), The data collection period is between July 2017 and December 2019. For each city, the data sample comprises a set of drivers for whom there were at least 50 trajectories longer than 30 seconds with valid GPS data available. For each driver, we ensured that the trajectories were collected during a relatively short period of time (e.g., two months). The last condition is to avoid any potential bias due to the time gap in the sampling process. For each trajectory, we find the origin and destination data points. Origin and destination points represented by GPS coordinates cannot be directly employed in our application. Thus, we use a reverse geocoding process to map-match origin and destination points to the closest places. The Nominatim API is used for the reverse geocoding process.

2) AHDC Study: The AHDC study is a representative, longitudinal study of 1,347 adolescents residing in Columbus, Ohio. The adolescents are provided with GPS-enabled smartphones and their movements are tracked through GPS for a period of 7 days. The details of the AHDC dataset are shown in Table I. Note that the nature of the AHDC dataset is different than that of Nationwide dataset in the following ways: 1. The individuals in AHDC study are adolescents rather than adults and 2. The GPS coordinates are collected from GPS-enabled mobile devices rather than automobiles. Another important aspect of this dataset is that it is carefully curated to mitigate noise effects of GPS positioning – specifically, visited location information is also independently verified by extrinsic sources (nightly surveys confirming each visited location).

B. Baselines and Evaluation Methodology

We evaluate the efficacy of the proposed method against baselines based on three criteria, namely, embedding quality, memory consumption and running time. We also report the results of our proposed LocationTrails model via Federated Learning. We select following baselines: Deepwalk [27], Node2vec [28], NetMF [29], LINE [31], BiNE [32], Deep Graph Infomax (DGI) [30]. All baselines operate on the co-location graph containing both user ids and location ids. We also tune the parameters of these methods and report their best performance. The grid-search parameters for Deepwalk, Node2vec, and Metapath2vec are windows = [2, 3, 5, 10], walk lengths = [20, 40], number of walks = [20, 40] and epochs = [20, 50, 100]. For Node2vec, we set p=4.0 and q=0.25. For Metapath2vec, the metapaths are: [['user', 'location', 'user'], ['location', 'user', 'location']]. NetMF parameters : windows = [5, 10, 20], ranks = [100, 200, 500], negatives = [3, 5, 10], LINE parameters: negatives = [3, 5, 10] and epochs = [100, 200, 500], M-NMF parameters: communities = [20, 50, 100], α = [0.05, 0.2], β = [0.05, 0.2], and λ = [0.2, 0.4], BiNE parameters: negatives =[1, 4, 10], window size = [1, 5, 9], probs = [0.05, 0.15, 0.5], β =[0.001, 0.01, 1] and γ = [0.01, 0.1, 5], DGI parameters: Dropout = [0.0, 0.25, 0.5] and weight decay = [0.0, 0.1, 0.2].

	Columbus	Atlanta	Pittsburgh	Philadelphia	AHDC Study
# Users	3,586	779	1,230	1,317	1,347
# Locations	41,630	23,757	22,990	30,971	5,572
Mode length	4	3	3	4	4
Mean length	4.61	4.07	4.16	4.64	4.33
# of Trails	159,860	71,254	120,619	133,898	6,483

TABLE I
THE STATISTICS OF THE TRAILS ON MULTIPLE DATASETS.

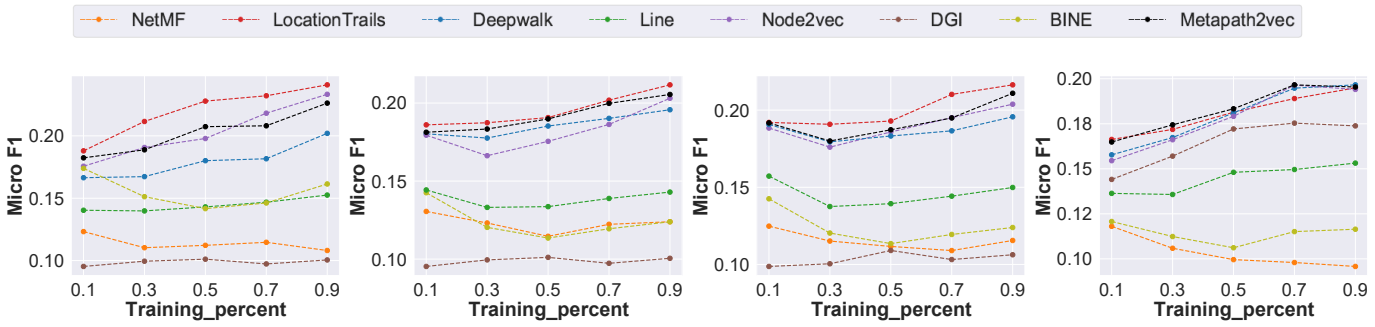
Note that the selected methods consist of recent network representation learning methods (DGI, BINE, NetMF) and also include the Deepwalk method which is found to be a competitive baseline [40], [41].

LocationTrails parameters: We perform grid search on following parameters: windows = [2, 3, 5, 10] and epochs = [20, 50, 100]. The number of negative samples is set to 5 and the embedding dimension is set to 128 for all the methods. Note that LocationTrails does not require either the number of walks or walk length as parametric input.

Evaluation of Embedding quality: In both use-case settings node classification is an important cue for qualitative performance so we compare the performance of LocationTrails and the competitive strawman on this specific task. Specifically, we evaluate the embedding quality through the location label classification task. Ground truth location labels for the Nationwide dataset is collected from the place "type" present in OpenstreetMaps place records. We consider only those place types whose number of instances in our dataset is smaller than 600 and greater than 50. We perform multi-class classification for the Nationwide dataset. The class imbalance problem is addressed through oversampling [42] of the minority class.

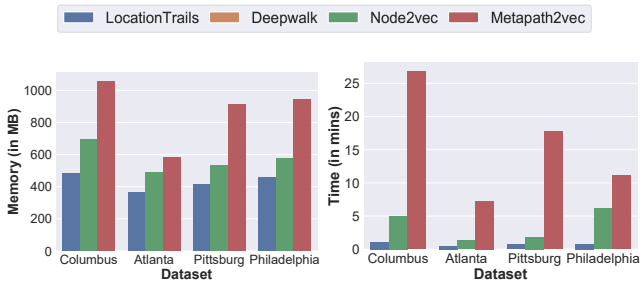
Ground truth location labels for AHDC study is collected through the survey performed in the study where adolescents provided semantic label of the visited location. However, a location could have a semantic label of "Home" for a particular adolescent while the same location could have a semantic label as "Friends' Home" for another adolescent. Hence we perform multi-label classification – instead of performing multi-class classification – on the AHDC dataset.

Experiment Settings: We lever logistic regression as a classifier for node classification. We use grid-search with 5-fold cross-validation for tuning the hyperparameters of logistic regression. We report results on the micro-f1 metric which is known to be sensitive to the imbalance of the classes.



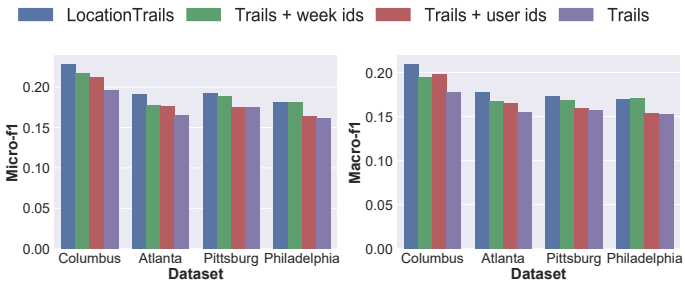
(a) Columbus (b) Atlanta (c) Pittsburgh (d) Philadelphia

Fig. 2. The performance of all the methods on multi-class classification task on the Nationwide datasets measured with Micro-f1 on various train:test splits.



(a) Memory (in MB) (b) Running time (in Mins)

Fig. 3. Memory consumption and running time on the Nationwide datasets.



(a) Micro-F1 (b) Macro-F1

Fig. 4. Ablation study on LocationTrails with three ablation factors.

C. Embedding Quality Comparison

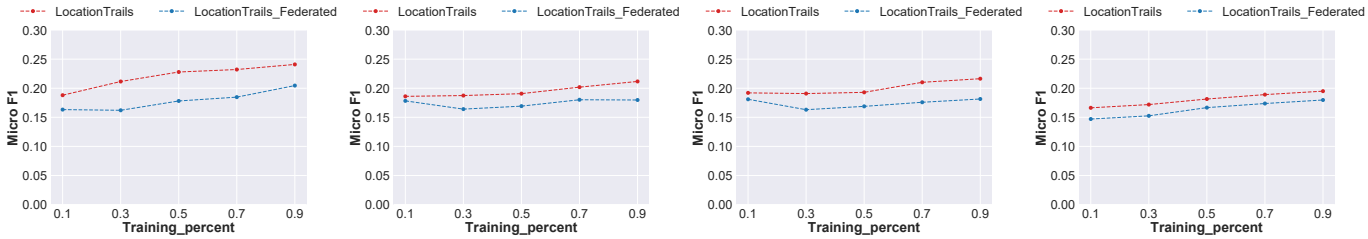
The location-label classification task results on micro-f1 is described in Figure 2 (Results are similar for the macro-f1 metric). The quality of the location embeddings learned by the LocationTrails model is often better than that of the competitive baselines. On all four datasets, we observe that the micro-f1 scores for LocationTrails, on the majority of the train: test splits, are better than that of the baselines. On all four datasets, we observe that the walk-based methods – LocationTrails, Deepwalk, Node2vec, and Metapath2vec – perform better than other baselines on the corresponding co-location networks. Often, the performance of Deepwalk, Node2vec, and Metapath2vec is closer to that of LocationTrails. However, as we will discuss in the next section, the memory requirement and running time cost of methods Deepwalk, Node2vec, and Metapath2vec are significantly higher than LocationTrails. Deep Graph Infomax (DGI) proposed for homogenous non-bipartite graphs doesn't seem to perform well on three co-location networks. On our largest dataset, Columbus, we observe that LocationTrails outperforms the next best baseline by 3% with micro-f1 metric on the train: test splits of 50:50. Additionally, on three datasets Columbus, Atlanta, and Pittsburgh, the classification performance of LocationTrails is better than all the baselines on most of the selected train: test splits in terms of micro-f1.

D. Ablation Study on Sequence Construction

The results of the ablation study on our proposed LocationTrails are shown in Figure 4. The ablation factors are *Trails* which consists of a sequence of locations, *Trails + user-ids* which consists of a sequence of location and user-ids interlaced together, *Trails + weekday/weekend ids* which consists of a sequence of location and weekday/weekend information interlaced together. We report the micro-f1 and macro-f1 scores of location-label classification with 50:50 train:test split. From Figure 4, we observe that LocationTrails location label classification performance is better than all other ablation factors on all the datasets except Philadelphia– *Trails + weekday/weekend ids* classification performance is slightly better than LocationTrails on macro-f1 metric. On micro-f1 metric, LocationTrails achieve up to 3.1% and 2.6% better classification performance as compared to *Trails* on the Columbus and Atlanta datasets, respectively.

E. Memory and Running time Comparison

The memory consumption in MB and running time in minutes of LocationTrails, Deepwalk, Node2vec, and Metapath2vec are shown in Figure 3. We select these methods because they are the most competitive ones. Additionally, these methods are implemented in the same language (python), and all of the methods lever the same gensim library [43]. We see that LocationTrails consumes less memory and learns location embedding much faster than random-walk based baselines. While reporting memory consumption and running time of the Node2vec method, *we also include the memory consumed by the Alias Table and the time taken to construct the Alias Table. This is because the construction of the Alias Table is required and necessary step for learning the node embeddings using Node2vec*. From Figure 3, we observe that LocationTrails requires 1.6x - 2.2x less memory than Node2vec and 1.32x - 1.44x less memory than Deepwalk. Additionally,



(a) Columbus (b) Atlanta (c) Pittsburgh (d) Philadelphia

Fig. 5. Classification performance of LocationTrails trained with stochastic gradient descent vs federated learning on Nationwide datasets.

LocationTrails is 7x - 25x times faster than Node2vec and 2.6x - 8x times faster than Deepwalk. The running time of Metapath2vec is highest because while performing a random walk, it has to select a neighboring node of a particular type based on the metapath schema which is time consuming. In short, LocationTrails requires less memory, learns location embedding significantly faster than Deepwalk, Node2vec and Metapath2vec and learns better quality location embeddings.

F. Federated Learning

Setup: We implement the LocationTrails-Federated model in the Tensorflow Federated framework. The training is performed on the simulated clients and model averaging is performed on the simulated server. We implement the simulated clients, training procedure and the simulated server using the Federated Core API present in the framework. The Tensorflow Federated – at the time of writing this paper – provides only single-machine simulation runtime; multi-machine simulation is not yet available to the public. In the framework, the global model is replicated on all the simulated clients, as a result, training LocationTrails with thousands of simulated clients is not feasible on our single machine with 28 cores and 128 GB RAM. We select the number of simulated clients equal to 20 and randomly hash a user to a client. Each client k trains the model w_t^k based on the location visit data of all the users hashed to that client. Note that the hashing function does not explicitly hash users residing in the same neighborhood to the same client. We keep the same set of hyperparameters for both the LocationTrails model trained with standard gradient descent and the LocationTrails model trained with Federated learning. The grid-search parameters for the total number of federated rounds is [20, 50, 100]. In this work, we focus on the unsupervised representation learning of existing users/locations – a common assumption in network representation learning. Modeling new users and locations is left for future work.

Figure 5 show the results on location label classification task with multiple train:test splits on Micro-f1[2] for: i) the LocationTrails model trained with standard gradient descent, and ii) the LocationTrails model trained with Federated Learning (LocationTrails_Federated), on four real-world human mobility datasets. The quality of embeddings learned by the LocationTrails model trained with Federated Learning is comparable to that of the LocationTrails model trained with standard gradient descent. On the Philadelphia dataset, we see around 1% performance difference between different training procedures while on Atlanta and Pittsburgh, we see around 2-3% performance difference with respect to Micro-f1. On Columbus dataset, the maximum micro-f1 difference is around 5% while minimum micro-f1 is around 2%. The drop in classification is not surprising in that similar performance has been observed in other works too [11], [15], where the training a model with standard stochastic gradient descent achieve a particular test accuracy, while the training the same model with Federated Learning achieve a slightly lower test accuracy. For example, the human activity recognition test accuracy of 93% versus 89% in [11] and the image classification test accuracy of 86% versus 85% in [15]. We reiterate that while there is a slight drop in classification performance, the federated approach comes with an additional advantage in user privacy.

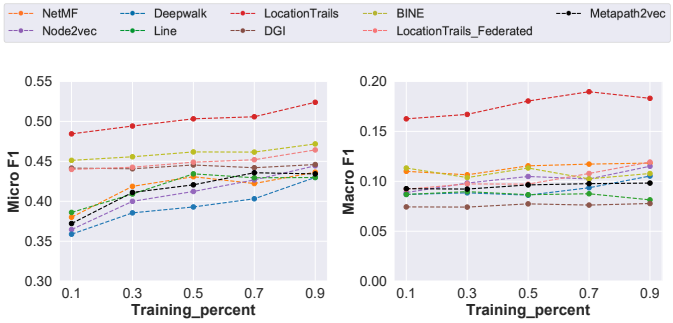
[2]Macro-f1 figure not shown due to the paucity of space. We observe similar trends with the Macro-f1 metric to that observed with the Micro-f1 metric.

G. AHDC Study

In this section, we show that our proposed LocationTrails model is a generalized model and can be applied in scenarios where human mobility data is collected from mobile phones. The multi-label location classification performance of the baselines is reported in Figure 6. We observe that the proposed LocationTrails model outperforms all other baselines on all the train: test splits. There is slight drop in the performance for LocationTrails model trained with Federated Learning - similar performance has been observed in other works too [11], [15]. We observe more than 7% absolute difference in macro-f1 score between LocationTrails on the best baseline with 90:10 train: test split. Additionally, we observe that as we increase the training percent, the rate of increase in location classification for LocationTrails is higher than that of the baselines. We also perform the ablation study along with memory and running time comparison of the methods on the AHDC dataset with the experimental setup for these experiments kept similar to that of conducted on the Nationwide dataset. We observe similar trends in AHDC study as observed in the Nationwide dataset in terms of ablation factors, memory consumption and execution time for the LocationTrails model.

V. Concluding Remarks

In this work, we propose LocationTrails a simple but novel model to efficiently learn meaningful location representations



(a) Micro-F1 (b) Macro-F1

Fig. 6. The performance of all the methods on multi-label classification task on the AHDC dataset.

from human mobility data. We then present a variation of the model whose training and subsequent model updates can happen on edge devices through a federated learning approach. We perform experiments on five real-world human mobility datasets and show the efficacy of our proposed approach w.r.t competitive baselines in terms of embedding quality, memory consumption, and execution time.

With respect to impact and deployment, we note that the ideas presented in this work are currently being integrated in the next wave of the AHDC study. We are also engaged with the insurance and mobility sector to see how such ideas can enhance the development of next-generation AI on the edge models for driver risk prediction. Specifically in this context, we are currently examining ways to quantify the benefits of such federated models, by also leveraging tools from differential privacy and secure aggregation to explore the trade-off between users privacy and location representation quality and end-utility.

Acknowledgment

This material is based upon work supported by the National Institute of Health under Grant No NIH-1R01HD088545-01A1 and National Science Foundation under OAC-2018627, CCF-2028944, and CNS-2112471. Any opinions, findings, and conclusions in this material are those of the author(s) and may not reflect the views of the respective funding agency.

References

[1] C. Ratti *et al.*, "Mobile landscapes: using location data from cell phones for urban analysis," in *Environment and planning B: Planning and design*, 2006.
[2] R. Ahas and Ü. Mark, "Location based services—new challenges for planning and public administration?" in *Futures*, 2005.
[3] R. J. Sampson *et al.*, "Neighborhoods and violent crime: A multilevel study of collective efficacy," in *Science*, 1997.
[4] H. Yin *et al.*, "Lcars: a location-content-aware recommender system," in *KDD*, 2013.
[5] B. Yan *et al.*, "From itdl to place2vec: Reasoning about place type similarity and relatedness by learning embeddings from augmented spatial contexts," in *SIGSPATIAL*, 2017.
[6] V. Spruyt, "Loc2vec: Learning location embeddings with triplet-loss networks," 2018.
[7] Vazquez-Prokopec *et al.*, "Using gps technology to quantify human mobility, dynamic contacts and infectious disease dynamics in a resource-poor urban environment," in *PloS one*, 2013.
[8] Y.-L. Lee, P.-K. Tsung, and M. Wu, "Technology trend of edge ai," in *VLSI-DAT*, 2018.
[9] J. Liang, S. Gurukar, and S. Parthasarathy, "Mile: A multi-level framework for scalable graph embedding," in *ICWSM*, 2021.
[10] European-Data-Protection-Board, *Guidelines 1/2020 on processing personal data in the context of connected vehicles and mobility related applications*, 2020.
[11] K. Sozinov *et al.*, "Human activity recognition using federated learning," in *ISPA*, 2018.
[12] M. J. Sheller *et al.*, "Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation," in *International MICCAI Brainlesion Workshop*, 2018.
[13] A. Hard *et al.*, "Federated learning for mobile keyboard prediction," in *arXiv preprint arXiv:1811.03604*, 2018.
[14] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NeurIPS*, 2013.
[15] H. B. McMahan *et al.*, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, 2017.
[16] S. Moosavi *et al.*, "Short and long-term pattern discovery over large-scale geo-spatiotemporal data," in *KDD*, 2019.
[17] S. Moosavi, B. Omidvar-Tehrani *et al.*, "Characterizing driving context from driver behavior," in *SIGSPATIAL*, 2017.
[18] Y. Shoji *et al.*, "Location2vec: Generating distributed representation of location by using geo-tagged microblog posts," in *ICSI*, 2018.
[19] D. Yang *et al.*, "Revisiting user mobility and social relationships in lbsns: A hypergraph embedding approach," in *TheWeb*, 2019.
[20] X. Liu *et al.*, "Exploring the context of locations for personalized location recommendations." in *IJCAI*, 2016.
[21] S. Zhao *et al.*, "Geo-teaser: Geo-temporal sequential embedding rank for point-of-interest recommendation," in *TheWeb*, 2017.
[22] B. Chang *et al.*, "Content-aware hierarchical point-of-interest embedding model for successive poi recommendation." in *IJCAI*, 2018.
[23] H. A. Rahmani *et al.*, "Category-aware location embedding for point-of-interest recommendation," in *SIGIR*, 2019.
[24] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.
[25] A. Vaswani *et al.*, "Attention is all you need," in *NeurIPS*, 2017.
[26] J. Devlin *et al.*, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.
[27] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *KDD*, 2014.
[28] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *KDD*, 2016.
[29] J. Qiu *et al.*, "Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec," in *WSDM*, 2018.
[30] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *ICLR*, 2019.
[31] J. Tang *et al.*, "Line: Large-scale information network embedding," in *TheWeb*, 2015.
[32] M. Gao, L. Chen, X. He, and A. Zhou, "Bine: Bipartite network embedding," in *SIGIR*, 2018.
[33] H. Cai *et al.*, "A comprehensive survey of graph embedding: Problems, techniques, and applications," in *TKDE*, 2018.
[34] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," in *NeurIPS*, 2017.
[35] D. Bui *et al.*, "Federated user representation learning," *arXiv preprint arXiv:1909.12535*, 2019.
[36] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *ICLR*, 2018.
[37] K. Bonawitz, V. Ivanov, B. Kreuter *et al.*, "Practical secure aggregation for privacy-preserving machine learning," in *SIGSAC*, 2017.
[38] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *ICML*, 2019.
[39] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *arXiv preprint arXiv:1807.00459*, 2018.
[40] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Systems*, 2018.
[41] S. Gurukar, P. Vijayan, A. Srinivasan, G. Bajaj, C. Cai, M. Keymanesh, S. Kumar, P. Maneriker, A. Mitra, V. Patel *et al.*, "Network representation learning: Consolidation and renewed bearing," *arXiv preprint arXiv:1905.00987*, 2019.
[42] G. Lemaître *et al.*, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," in *JMLR*, 2017.
[43] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *LREC*, 2010.