Beware of What You Share: Inferring User Locations in Venmo

Xin Yao, Member, IEEE, Yimin Chen, Student Member, IEEE, Rui Zhang, Member, IEEE, Yanchao Zhang, Senior Member, IEEE, and Yaping Lin, Member, IEEE

Abstract—Mobile payment apps are seeing explosive usage worldwide. This paper focuses on Venmo, a very popular mobile person-to-person payment service owned by PayPal. Venmo allows money transfers between users with a mandatory transaction note. More than half of transaction records in Venmo are public information. In this paper, we propose a multilayer location inference (MLLI) technique to infer user locations from public transaction records in Venmo. MLLI explores two observations. First, many Venmo transaction notes contain implicit location cues. Second, the types and temporal patterns of user transactions have strong ties to their location closeness. With a large dataset of 2.12M users and 20.23M Venmo transaction records, we show that MLLI can identify the top-1, top-3, and top-5 possible locations for a Venmo user with accuracy up to 50%, 80%, and 90%, respectively. Our results highlight the danger of sharing transaction notes on Venmo or similar mobile payment apps.

Index Terms—Location inference, mobile payment, privacy, security.

I. INTRODUCTION

MOBILE payment apps are seeing explosive usage worldwide. According to [1], the volume of mobile payment transactions could rise from $25B in 2016 to nearly $275B by 2021, amounting to an average annual growth rate of 62%. As another example, Alipay and WeChatPay, two popular payment systems in China, have 100M daily active users in December 2016 and 200M users in January 2016, respectively.

This paper focuses on Venmo, a very popular mobile person-to-person (P2P) payment service owned by PayPal. Venmo had 203 million active users and processed $6.8 billion in payment volume in Q1 2017 [2]. It is essentially a combination of social and transaction networks. On the one hand, Venmo allows users to befriend each other as in Facebook-like online social networks (OSNs). On the other hand, it allows money transfers between users by phone numbers, Venmo usernames, or e-mails along with a mandatory transaction note in the form of words, emojis, or their combinations. For example, Alice paid Bob $5.1 for pizza or charged Bob $550 for 🍬. In Venmo, users can make their transaction records viewable by the public with the default system setting or by selected parties only via privacy control. As reported in [3], almost half of Venmo transaction records are public information.

This paper presents the first systematic study to infer the home locations of Venmo users from public transaction records. We follow the convention [4]–[15] to let a home location (or location for short) refer to a permanent and static city-level region, where most of the user’s daily activities occur. This paper may have significant positive and negative impacts. On the positive side, the knowledge about user locations may benefit many applications, such as socioeconomic studies, local event recommendation, and business promotions. On the other hand, the disclosure of home locations may subject the users to many attacks such as location-based spam campaigns. This paper is also very challenging because explicit location clues are relatively sparse and often unreliable in public Venmo transaction records. For example, our large-scale dataset reveals that only 13.34% of public transaction records contain geotagged information (city or state names in USA), most of which are unrelated to the home locations of Venmo users.

In this paper, we propose a multilayer location inference (MLLI) technique to infer user locations in Venmo. The design of MLLI is driven by two observations. First, many Venmo transaction notes contain implicit location cues. Second, the types and temporal patterns of user transactions have strong ties to their location closeness. For example, if David and Bob split lunch bills on a daily basis via Venmo, they are in the same city (i.e., have the same location) with overwhelming probability. In contrast, if they split monthly wireless bills via Venmo, there is relatively low confidence that they are in the same location because people far away from each other can still share a wireless plan.

MLLI explores the above observations in four steps. First, we use text mining algorithms to obtain the keywords for each transaction note. Since distinct keywords have different probabilities of occurring in the same location because people far away from each other can still share a wireless plan.
location relevance, we further divide the keywords and the corresponding transaction records into four categories, where the lower-numbered category corresponds to higher location relevance. Second, we construct an undirected weighted trans-
action graph for each category, in which each edge corresponds to two users with any transaction history in that category,
and the edge weight depends on their transaction pattern.
For example, more intense and consistent transactions should translate into higher edge weights than occasional ones. Third, we identify a small set of users as seeds whose locations can be directly obtained from their geotagged Venmo transaction notes or via external means.\(^1\) Then we propose an iterative
procedure. Let \(\mathcal{G} = (\mathcal{N}, \mathcal{E})\), where \(\mathcal{N}\) and \(\mathcal{E}\) refer to the user set and edge set, respectively. An edge \(e_{ij} \in \mathcal{E}\) is formed between any users \(i\) and \(j\) as long as they have historical transactions in Venmo. Who is the payment sender or receiver of any transaction has no bearing on our scheme design, so we ignore the directions of Venmo transactions. A set of transaction records associated with each edge \(e_{ij}\), we use text mining approaches to infer a keyword for each transaction record, referred to as its transaction purpose. Then we can group the transactions of the same purpose into a time series. For example, we can have \{Dinner: [01/14/2017@6:00pm, 01/17/2017@5:30pm]; Split: [02/27/2017@9:00am]\} associated with edge \(e_{ij}\), meaning that users \(i\) and \(j\) split dinner expenses twice and rent once at the corresponding time. Let \(N^+ \subseteq \mathcal{N}\) denote the set of users with known home locations (called seed users) and \(N^+ \cap \mathcal{N}\) denote the remaining users with unknown home locations (called nonseed users). So we have \(\mathcal{N} = N^+ \cup N^+\). We aim to tackle the following problem. Let \(\hat{h}_i\) for each nonseed user \(i \in N^+\) as close to its true location \(h_i\) as possible.

**Location Inference in Venmo:** Given a Venmo transaction graph \(\mathcal{G} = (\mathcal{N}, \mathcal{E})\) with an observed location \(h_j\) for each seed user \(j \in N^+\), estimate top-\(k\) possible home locations \(\hat{h}_i\) for each nonseed user \(i \in N^+\). Let \(N^-\) be the number of nonseed users whose true location \(h_i\) located at \(\hat{h}_i\). We aim to make \(|N^-|\) as close to \(|N^+|\) as possible.


### III. DATA CRAWLING

In this section, we introduce our data crawling process and the ground-truth dataset.

**A. Data Collection**

We used the public Venmo v5 API to retrieve historical transaction records in Venmo. Each retrieval request is a URL with a constant field and a Unix timestamp. As an example, the request is https://venmo.com/api/v5/public?until=1488369600 when the time is set to 03/01/2017@12:00pm (UTC). Venmo returns the most recent 20 records before the specified time in each retrieval request. By sweeping through the timestamps from December 24, 2016 to March 24, 2017, we obtained 8.2M unique users and 37.46M transactions in total. For our purpose, each transaction record contains the transaction initiator and receiver along with their public profiles, the transaction time, and the mandatory transaction note. Other information returned by Venmo is ignored. Our technique applies to the users with sufficient transactions. In this paper, we only consider users with more than ten transaction records in the crawled dataset, which correspond to 2.12M users with public information by default. A user can make his/her transaction records viewable by selected parties only as well. The recent study [3] reveals that almost half of Venmo transaction records are public information that can be easily obtained by invoking Venmo APIs.

We explore public Venmo transaction records to construct an undirected weighted transaction graph \(\mathcal{G} = (\mathcal{N}, \mathcal{E})\), where \(\mathcal{N}\) and \(\mathcal{E}\) refer to the user set and edge set, respectively. An edge \(e_{ij} \in \mathcal{E}\) is formed between any users \(i\) and \(j\) as long as they have historical transactions in Venmo. Who is the payment sender or receiver of any transaction has no bearing on our scheme design, so we ignore the directions of Venmo transactions. A set of transaction records associated with each edge \(e_{ij}\), we use text mining algorithms to infer a keyword for each transaction record, referred to as its transaction purpose. Then we can group the transactions of the same purpose into a time series. For example, we can have \{Dinner: [01/14/2017@6:00pm, 01/17/2017@5:30pm]; Split: [02/27/2017@9:00am]\} associated with edge \(e_{ij}\), meaning that users \(i\) and \(j\) split dinner expenses twice and rent once at the corresponding time. Let \(N^+ \subseteq \mathcal{N}\) denote the set of users with known home locations (called seed users) and \(N^+ \cap \mathcal{N}\) denote the remaining users with unknown home locations (called nonseed users). So we have \(\mathcal{N} = N^+ \cup N^+\). We aim to tackle the following problem. Let \(\hat{h}_i\) for each nonseed user \(i \in N^+\) as close to its true location \(h_i\) as possible.

**Location Inference in Venmo:** Given a Venmo transaction graph \(\mathcal{G} = (\mathcal{N}, \mathcal{E})\) with an observed location \(h_j\) for each seed user \(j \in N^+\), estimate top-\(k\) possible home locations \(\hat{h}_i\) for each nonseed user \(i \in N^+\). Let \(N^-\) be the number of nonseed users whose true location \(h_i\) located at \(\hat{h}_i\). We aim to make \(|N^-|\) as close to \(|N^+|\) as possible.

### III. DATA CRAWLING

In this section, we introduce our data crawling process and the ground-truth dataset.

**A. Data Collection**

We used the public Venmo v5 API to retrieve historical transaction records in Venmo. Each retrieval request is a URL with a constant field and a Unix timestamp. As an example, the request is https://venmo.com/api/v5/public?until=1488369600 when the time is set to 03/01/2017@12:00pm (UTC). Venmo returns the most recent 20 records before the specified time in each retrieval request. By sweeping through the timestamps from December 24, 2016 to March 24, 2017, we obtained 8.2M unique users and 37.46M transactions in total. For our purpose, each transaction record contains the transaction initiator and receiver along with their public profiles, the transaction time, and the mandatory transaction note. Other information returned by Venmo is ignored. Our technique applies to the users with sufficient transactions. In this paper, we only consider users with more than ten transaction records in the crawled dataset, which correspond to 2.12M users with

---

\(^1\)Some Venmo users link their Venmo accounts to Facebook accounts and disclose their locations on Facebook.
20.23M transaction records in total. During the data collection, we limited the query rates to avoid disruption to Venmo’s services. Although the transaction records in our dataset is publicly accessible via the Venmo v5 API, we will not share the dataset with others unless a written consent from Venmo can be obtained.

B. Ground Truth

It is critical to obtain a ground-truth dataset for our algorithm execution and performance evaluation. About 6% of Facebook users have elected to enter their home addresses in their public profiles [4]. Since a user can register for Venmo with his/her Facebook credentials, we used the following process to crawl the ground truth. We first identified the users whose profiles contain firstname+lastname as well as a user image or a friend list from the crawled transaction records. Then we searched the firstname+lastname for each such Venmo user on Facebook. Any discovered Facebook user is considered the same person as the Venmo user if they two have the same image or a highly similar friend list. If the Facebook user specifies the home location in the profile, the home location of the Venmo user is considered known, corresponding to one piece of ground truth. With this process, we obtained a ground-truth dataset of 1000 users from our original dataset.

IV. Categorized Transaction Graphs

Our MLLI technique explores the observation that public transaction notes in Venmo contain implicit cues for accurate location inference. In this section, we first extract meaningful keywords from the crawled transaction notes. Since different keywords vary in location relevance, we further group the keywords into four categories whereby to construct four categorized transaction graphs for subsequent use. Note that we do not harness explicit location cues in transaction notes, which are not only rare but also unrelated to home locations.

A. Keyword Extraction

We use text mining techniques to extract meaningful keywords from crawled transaction notes. For the words in each transaction note, we first remove the stop words using a stop-word list, in which such words as “the” and “those” are considered too general and meaningless. People may use different inflected words in their transaction notes. So we further conduct stemming [16] to reduce inflected words to their stems. For example, “play,” “playing,” and “played” are all reduced to play. Each transaction note is translated into its purified version after these two steps. All the words or emojis in a purified transaction note are considered its keywords. Fig. 1(a) shows the length distribution of 20.23M purified transaction notes involving 2.12M Venmo users. As we can see, more than 62% of transaction notes have one keyword, and about 20% have two keywords. This result provides firm evidence that the keywords can well characterize Venmo transaction notes. Table I shows the frequencies of the top-10 common keywords. We also draw the distribution of keyword frequency in Fig. 1(b), which shows that more than 75% of the keywords appeared only once.

B. Keyword Categorization

Different transaction keywords vary in relevance to user locations. For example, “rent” is a more reliable indicator than that the two users are in the same city (or location). Armed with this observation, we categorize the transaction keywords into four categories such that lower-numbered categories have higher location relevance than higher-numbered ones.

The keywords in category 1 and category 2 relate to physical or social activities that two users must conduct face to face. For example, rent and “electricity” are the strongest indicators that the two users involved live together and thus have the same location, so both belong to category 1. In contrast, “movie” and “party” correspond to face-to-face activities as well, but they are classified into category 2 because they are less stronger co-location indicators than rent.

The keywords in category 3 correspond to the activities that can happen face to face or not. For example, a transaction with the “gift” keyword can involve two users who split the gift cost for a common friend. The two users may live in the same city or not, so gift has weaker co-location relevance than movie and is assigned to category 3. In contrast, the keywords in category 4 are not explicitly correlated with locations.

We show some examples for each category below to help understand our categorization rule.

1) Category 1: Apartment, rent, electricity, cleaning, etc.
2) Category 2: Food, party, movie, uber, gas, etc.
3) Category 3: Gifts, clothing, music, medical, etc.
4) Category 4: Airfare, game, phone, insurance, etc.

We have obtained 493, 314, 356, and 337 keywords in category 1, category 2, category 3, and category 4, respectively. Recall that our crawled dataset comprises 2.12M users and 20.23M transaction records, where each user has at least ten transaction records. We further classify the transaction
methods to derive the edge weights. A transaction record can belong to multiple subsets if it has multiple keywords in different categories. Those without any categorized keyword do not belong to any subset and will not be further considered in our algorithm. Finally, we have subset 1 with 1.83M users and 4.24M transaction records, subset 3 with 1.83M users and 4.07M transaction records, and subset 4 with 1.83M users and 4.11M transaction records. In addition, we have 87.7% of the transactions in one subset, 11.1% in two subsets, 1% in three subsets, and only 0.07% in all the four subsets.

One limitation of our method is that involves manual classification. While it is feasible in our case, we left automating the process as our future work, which would need incorporate effective natural language processing techniques. In addition, our categorization method above is empirical and exemplary, and a finer classification of the transaction keywords can be adopted as well.

We also explore the ground-truth dataset to derive a co-location coefficient for each transaction subset. For each subset, we count the number of transaction records between any two users in the ground-truth dataset as well as those between any two users in the ground-truth dataset and also the same city (location). The co-location coefficient for this transaction subset is the later count divided by the former. Subset 1, subset 2, subset 3, and subset 4 have the co-location coefficients of 0.823, 0.739, 0.684, and 0.588, respectively. These results validate our assumption that the keywords in lower-numbered keyword categories are more reliable indicators that two transaction users are in the same location.

C. Construction of Categorized Transaction Graphs

The same keyword may have very different co-location relevance to different pairs of users. Intuitively speaking, more consistent transactions of the same keyword should translate into stronger location closeness. For example, Tom and Bob live in different cities and split a drink bill occasionally when they attend the same conference. In contrast, Tom and Jerry are colleagues and often split after-hour drink bills. The "drink" keyword is obviously more relevant between Tom and Jerry than between Tom and Bob.

To capture this observation, we construct an undirected weighted transaction graph $G_z = (\mathcal{N}_z, \mathcal{E}_z)$ for each category $z$ based on the transaction subset $z$, $\forall z \in [1, 4]$. The node set $\mathcal{N}_z$ includes all the users in subset $z$, and an edge $e_{i,j,z} \in \mathcal{E}_z$ exists if subset $z$ contains transaction records between users $i$ and $j$. Each edge $e_{i,j,z}$ has a weight $w_{i,j,z}$ that models the location closeness between users $i$ and $j$. We use the following methods to derive the edge weights.

1) Sum-Based: In this method, $w_{i,j,z}$ equals the total number of transaction records between users $i$ and $j$ in the transaction subset $z$.

2) Entropy-Based: In this method, we divide time into epochs of the same length (say, one week). Let $d_{i,j,z}^{(i,j)}$ denote the number of transactions in subset $z$ between users $i$ and $j$ in epoch $x$. We define

$$w_{i,j,z} = \left(1 - \sum_x \frac{d_{i,j,z}^{(i,j)}}{\sum_x d_{i,j,z}^{(i,j)}} \cdot \log \frac{d_{i,j,z}^{(i,j)}}{\sum_x d_{i,j,z}^{(i,j)}}\right) \cdot \sum_x d_{i,j,z}^{(i,j)}.$$

Neither method above is perfect. In particular, the sum-based method translates more transactions into higher edge weights (location closeness), but it does not capture the temporal information in transaction records. Continue with the previous Tom–Bob–Jerry example. Tom and Bob may have a lot of Venmo transactions in a short period (say, one week) while attending the same conference. But Tom and Jerry may have one Venmo transaction every week in the past month. The sum-based method will give a wrong co-location indication in this scenario. In contrast, the entropy-based method can produce a much higher edge weight between Tom and Jerry than between Tom and Bob, but it cannot reflect the transaction volume.

V. MLLI

In this section, we present our MLLI technique to infer the locations of Venmo users from categorized transaction graphs $\{G_z\}_{z=1}^4$.

We use the following key notation. Assume that there is an ordered list of all possible cities (locations). For example, there are 307 cities with more than 100K residents in the United States in 2016. Let $p_i$ denote a vectorized location belief for each user $i$, where the $k$th element $p_i(k)$ denotes the probability that user $i$ is in the $k$th city in a predefined list. $p_i$ is inferred from the combination of $\{G_z\}_{z=1}^4$. We also use $p_{i,z}$ to denote user $i$’s location belief inferred from $G_z$ alone, whose $k$th element is denoted by $p_{i,z}(k)$.

MLLI derives $p_i$ for each user $i$ in three steps. First, we partition each transaction graph $G_z$ into communities. Second, we select a few seed users in each community whose location beliefs can be known a priori, and then we use the max-product belief propagation (MPBP) technique to iteratively propagate the location beliefs inside each community. Finally, we derive $p_i$ as a weighted combination of the individual location beliefs $\{p_{i,z}\}_{z=1}^4$ in an iterative fashion.

A. Community Division

The users in the same city can be expected to have more intense Venmo transactions with each other than with others in different cities, leading to a strongly connected community in each categorized transaction graph $G_z$. The community structure in a large undirected graph can be inferred by maximizing the modularity [17]. We adopt the Louvain method [18], [19], a popular modularity-based technique to divide each $G_z$ into communities. The Louvain method is a greedy optimization method that attempts to optimize the modularity of a partition in the graph, which is defined as a value between -1 and 1 that measures the density of links inside communities in contrast to those between communities. In [20], the modularity of each graph $G_z = (\mathcal{N}_z, \mathcal{E}_z)$ is defined as

$$Q_z = \frac{1}{2 \cdot m_z} \sum_{i,j} \left[ w_{i,j,z} - \frac{\sum_j w_{i,j,z} \cdot \sum_i w_{i,j,z}}{2 \cdot m_z} \right] \cdot \delta(p_{i,z}, p_{j,z}).$$
where $w_{i,j,z}$ denotes the edge weight between users $i$ and $j$, $m_z = (1/2) \sum_{i,j} w_{i,j,z}$ is the total edge weight in $G_z$, $\sum_{i,j} w_{i,j,z}$ is the total edge weight concerning user $i$, $\rho_{i,z}$ and $\rho_{j,z}$ refer to the community indexes of nodes $i$ and $j$, respectively, and $\delta(\rho_{i,z}, \rho_{j,z})$ is a Kronecker delta function. In particular,

$$\delta(\rho_{i,z}, \rho_{j,z}) = \begin{cases} 1, & \rho_{i,z} \neq \rho_{j,z} \\ 0, & \rho_{i,z} = \rho_{j,z}. \end{cases}$$

We explain the intuition of the modularity as follows. Considering any two users $i$ and $j$, we utilize $(\sum_{j} w_{i,j,z} \cdot \sum_{i} w_{i,j,z})/2 \cdot m_z$ to denote the expectation that they can form an edge in graph $G_z$. If their communities are not the same (i.e., $\rho_{i,z} \neq \rho_{j,z}$), they have no contribution to $Q_z$. If $i$ and $j$ belong to the same community (i.e., $\rho_{i,z} = \rho_{j,z}$), they have positive impact on $Q_z$ if they are neighbors and negative impact otherwise.

With the Louvain method, we obtain a community set $C_z$ for each graph $G_z$. Table II lists the numbers of communities with the modularity in each graph $G_z$. According to [21], any modularity greater than 0.3 indicates meaningful community structures. So our four categorized transaction graphs all have very meaningful community structures, with 95.51%, 93.79%, 94.86%, and 94.59% of the transactions taking place within communities, respectively.

## B. Max-Product Location-Belief Propagation

Recall our conjecture that user transactions in Venmo exhibit strong geographic locality in the sense that users in the same area tend to have more intensive transactions with each other than with those from outside. Therefore, after obtaining the community set $C_z$ for each graph $G_z$, we can expect that the location of each user can be inferred from those of others in the same community. The locations of some users may be known a priori, e.g., by using the same method for constructing the ground-truth dataset (see Section III-B). These users are referred to as seed users whose location beliefs are accordingly known as well. For example, if seed $s$ is in the $k$th city, the location belief $p_s$ has 1.0 in the $k$th position and 0s in all the other positions. For each seed user, we apply belief propagation techniques to propagate the location beliefs within its community to evaluate the likelihood that all the other users in the community are in the same location (city).

We use the classical MPBP technique [22], [23] as an example for location-belief propagation, and many other belief propagation techniques can be applied as well. We rename the technique max-product location-belief propagation (MP-LBP). To avoid introducing a new set of notation, we assume that graph $G_z$ corresponds to a single community where one or more seeds are in the $k$th city. Each seed $s$ has its $k$th location-belief element (probability) $p_s(k) = 1.0$. In contrast, every nonseed user $i \in G_z$, has $p_i(k) = 0.5$ initially, corresponding to the equal probability that user $i$ is in the $k$th city or not. We follow [24] to model $G_z$ as a pairwise Markov random field (PMRF). Consider any two neighbors $i$ and $j$ in $G_z$. Let $x_i(k)$ be a binary indicator about whether user $i$ lives in the $k$th city: $x_i(k) = 1$ if so and $x_i(k) = 0$ otherwise. MP-LBP is an iterative process. Let $p_i^{(0)}(k)$ denote the result in iteration $t \geq 0$, where $p_i^{(0)}(k) = 0.5$. We also define $g_i(k) = p_i^{(t-1)}(k)$ $(\forall t \geq 1)$, i.e., the prior probability that user $i$ is in the $k$th city before iteration $t$. We further define a node potential for each node $i$ as

$$\theta_i(k) = \begin{cases} g_i(k), & \text{if } x_i(k) = 1 \\ 1 - g_i(k), & \text{if } x_i(k) = 0 \end{cases}$$

and an edge potential for each edge $e_{i,j}$ as

$$\psi_{i,j}(k) = \begin{cases} \mathcal{J}_{i,j}(k), & \text{if } x_i(k) = x_j(k) \\ 1 - \mathcal{J}_{i,j}(k), & \text{if } x_i(k) \neq x_j(k). \end{cases}$$

Here $\mathcal{J}_{i,j}(k) = (1 + \exp(-w_{i,j,z}/d_{i,z}))^{-1}$ and $d_{i,z} = \sum_{j} w_{i,j,z}$ denotes the total edge weight concerning user $i$ in graph $G_z$. Note that the node potential $\theta_i(k)$ represents the possibility that whether user $i$ lives in the $k$th city or not, and the edge potential $\psi_{i,j}(k)$ denotes correlations between $x_i(k)$ and $x_j(k)$ in the graph $G_z$.

In each iteration of MP-LBP, each node receives messages from its neighbors simultaneously, then updates its location belief, and finally sends a new message to each neighbor in the end of the iteration. For any two neighboring nodes $i$ and $j$ in $G_z$, we define message $m_{i,j}^{(0)}(x_i(k))$ as the influence that the state of node $j$ in the $k$th city [i.e., $x_j(k)$] has on user $i$’s location belief in the $t$th iteration ($t \geq 0$). Following the prior work [25], we set $m_{i,j}^{(0)} = 0.5$ for any two neighbors $i,j$. MPBP iteratively updates each message as

$$m_{i,j}^{(t)}(x_i(k)) = \max_{x_i(k)} \psi_{i,j}(k) \cdot \theta_i(k) \cdot \prod_{u \in \Gamma_{i,j}} m_{u,i}^{(t-1)}(x_i(k))$$

where $\Gamma_{i,j}$ means all neighbors of user $i$ except user $j$. Equation (1) means that we always select the most likely result between user $i$’s two possible states for city $k$, i.e., $x_i(k) = 1$ and $x_i(k) = 0$. MPBP repeats until the messages become negligible in two consecutive iterations (e.g., the $l_1$ distance of changes becomes smaller than $10^{-3}$), or it reaches the predefined maximum number of iterations. After convergence in iteration $t$, we estimate the belief $p_i^{(t)}(k)$ [denoted by $p_i^{(t)}$] as

$$p_i^{(t)}(k) \cdot \prod_{u \in \Gamma_{i}} m_{u,i}^{(t)}(k) \cdot \prod_{u \in \Gamma_{i}} \left( 1 - m_{u,i}^{(t)}(k) \right) \cdot \prod_{u \in \Gamma_{i}} \left( 1 - m_{u,i}^{(t)}(k) \right) .$$

The above iterative process is run in every community of each graph $G_z$ ($\forall z \in [1, 4]$) where a seed user exists. If there are multiple seeds in one community, a single run is done per each unique seed location, in which all the seeds in the same location are simultaneously involved. Each user $i$ in $G_z$ belongs

<table>
<thead>
<tr>
<th>Graphs</th>
<th># of communities</th>
<th>Modularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G_1^5$</td>
<td>130029</td>
<td>0.95</td>
</tr>
<tr>
<td>$G_2^3$</td>
<td>109344</td>
<td>0.93</td>
</tr>
<tr>
<td>$G_3^3$</td>
<td>126181</td>
<td>0.94</td>
</tr>
<tr>
<td>$G_4^4$</td>
<td>125665</td>
<td>0.93</td>
</tr>
</tbody>
</table>
Algorithm 1: ML-LBP

Input: Community set \{C_z\}_{z=1}^4, \ L
Output: Location belief \(p_i\) for each user \(i\)
1 Initialize \(p_{i,z}^{(0)}\) for each user \(i\), \(\forall z \in [1, 4]\);
2 \(p_i^{(0)} \leftarrow \sum_{z \in [1, 4]} \beta_z \cdot p_{i,z}^{(0)}\);
3 \(l \leftarrow 1\);
4 while \(l \leq L\) do
5 for each community set \(C_z\) do
6 Compute \(p_i^{(l)}\) by MP-LBP with \(p_{i,z}^{(l-1)}\) as the input for each community in \(C_z\);
7 \(p_i^{(l)} \leftarrow \sum_{z \in [1, 4]} \beta_z \cdot p_{i,z}^{(l)}\), \(\forall i;\)
8 if \(\|p_i^{(l)} - p_i^{(l-1)}\|_1 \leq 10^{-3}, \forall i\) then
9 \(\text{return } p_i^{(l)}, \forall i\)
10 \(l \leftarrow l + 1;\)
11 \(\text{return } p_i^{(l)}, \forall i\)

Intuitively, the more seeds we have, the more distributed the seeds are to different communities, the more nonseed users that can be profiled. In practice, it involves nontrivial effort to discover trustworthy seed users with the same method for constructing the ground-truth dataset (see Section III-B). So we are interested in the minimum of seeds needed to reach the given coverage-ratio \(\lambda\) such that the percentage of profiled users is no smaller than \(\gamma\). Alternatively, given a budget on the number of seeds, how should we select the seeds to maximize the coverage-ratio?

Seed selection can translate into the conventional maximum coverage problem (MCP). In particular, recall that \{\(C_1, C_2, C_3, C_4\)\} denote all the possible communities in the four categories, and the communities at different categories may include the same user who has transaction records in multiple categories. We abuse the notation by letting each \(C_z\) (\(\forall z \in [1, 4]\)) denote the set of users there as well. So we have \(\bigcup_{z=1}^4 C_z\) equivalent to the node set \(N\) in the original transaction graph \(G = (N, E)\). Let \(C'\) denote an arbitrary set of communities in \(\{C_1, C_2, C_3, C_4\}\), i.e., \(C' \subseteq \{C_1, C_2, C_3, C_4\}\). Seed selection can be casted as the following problem.

Definition 1 (Seed Selection): Given a coverage ratio \(\lambda\) and a set family \(\{C_1, C_2, C_3, C_4\}\), find a minimum-size subset \(C' \subseteq \{C_1, C_2, C_3, C_4\}\) such that \(\big|\bigcup_{C \in C'} C\big|/\big|\bigcup_{z=1}^4 C_z\big| \geq \lambda\).

The above definition assumes that no more than one seed is chosen in each community to maximize the coverage ratio for a given number of seeds. Seed selection is equivalent to the classical MCP which is known to be NP-hard. We address seed selection with the generic greedy algorithm [26], [27] which can approximate MCP in the best-possible polynomial time. Fig. 2(a) depicts the number of seeds versus the coverage ratio \(\lambda\).

The generic greedy algorithm only outputs the communities for a given \(\lambda\), but we may not be able to discover a seed user in each such community. If this happens, we consider the corresponding community nonprofitable and run the generic greedy algorithm again by ignoring it. Our experiments indicate that this situation is rare. Fig. 2(b) depicts the percentage of communities with at least one seed, which is over 90% for different coverage ratios.

VI. THEORETICAL ANALYSIS

A. Convergence Analysis

In this section, we analyze the convergence of MP-LBP and ML-LBP. MP-LBP is a linearized MPBP scheme over a
pMRF. Following the work in [25] and [28], we linearize (2) with two steps. In the first step, when user \( i \) sends a message to his/her neighbor \( j \), we need to consider the message that his/her neighbor \( j \) sends to \( i \) instead of excluding this message. In this case, we have the following equation:

\[
\mathbf{m}^{(t)}_{i,j,z}(x_i(k)) = \max_{x_k(k)} \phi_i,j,z(k) \cdot \mathbf{p}^{(t-1)}(x_i(k)).
\]

In the second step, we define the residual values \( \tilde{\mathbf{p}}^{(t)}_{i,z}(k) \) and \( \tilde{g}_i,z(k) \) as \( \mathbf{p}^{(t)}_{i,z}(k) - 0.5 \) and \( \mathbf{m}_i^{(t)}(k) - 0.5 \), respectively. Based on these notation, we replace (2) by

\[
\tilde{\mathbf{p}}^{(t)}_{i,z}(k) = \tilde{g}_i,z(k) + \sum_{u \in \Gamma_i} \tilde{\mathbf{m}}^{(t)}_{u,i,z}(k).
\]

The proof can be found in [25, Appendix A]. Incorporating (4) into (3) yields two cases: if \( x_i(k) = 1 \), \( \tilde{\mathbf{p}}^{(t)}_{i,z}(k) = \tilde{g}_i(z(k) + V_i \cdot (\phi_i,j,z(k) + \tilde{\mathbf{p}}^{(t-1)}_{i,z}(k)) + 0.5 - 0.5) \); if \( x_i(k) = 0 \), \( \tilde{\mathbf{p}}^{(t)}_{i,z}(k) = \tilde{g}_i(z(k) + V_i \cdot ((1 - \phi_i,j,z(k)) \cdot \tilde{\mathbf{p}}^{(t-1)}_{i,z}(k) - 0.5). \) Note that \( V_i \) denotes the \( i \)th row in the adjacency matrix \( \mathbf{M} \) of the graph \( G_t \). To simplify the analysis, we set \( \phi_i,j,z(k) = \mathcal{J} \) for any two users [25]. According to [29], the convergence condition for an iterative linear process \( \mathbf{y}^{(0)} \rightarrow -c + \mathbf{M}^{y^{(0)-1}} \) is that the spectral radius of the matrix \( \mathbf{M} \) [i.e., \( \rho(\mathbf{M}) \)] is no larger than 1. Therefore, the sufficient and necessary condition that the linearized MP-LBP converges is \( \mathcal{J} < (1/\rho(\mathbf{M})) \). ML-LBP is more complicated and involves many calls of MP-LBP. We could not guarantee whether ML-LBP will converge. When applying ML-LBP in practice, we terminate it empirically until there is no more significant change in the location beliefs in two consecutive iterations, or a maximum tolerable number of iterations are reached.

B. Complexity Analysis

In this paper, the computational time of MLLI includes the time for community division and time for ML-LBP. For a graph \( \mathcal{G} = (\mathcal{V}, \mathcal{E}) \), the exact time complexity of community division is unknown, but we can run the Louvain method in time \( \mathcal{O}(|\mathcal{V}| \log |\mathcal{V}|) \) [20]. Since we deploy the Louvain method to four graphs and obtain their corresponding communities, the time complexity of community division in our scheme is \( \mathcal{O}(|\mathcal{V}| \log |\mathcal{V}|) \). As discussed in [23], the computation complexity of MP-LBP for the graph \( \mathcal{G} \) is \( \mathcal{O}(t \cdot |\mathcal{E}| \cdot \sigma) \), where \( t \) is the number of iterations in MP-LBP, and \( \sigma \) is the number of unique locations. In ML-LBP, MP-LBP is applied to all the graphs in the community set \{\( C_1, C_2, C_3, C_4 \)\} in each iteration, so the time complexity of ML-LBP is \( \mathcal{O}(t \cdot |V| \cdot |\mathcal{E}| \cdot \sigma) \). Here, \( |V| \) denotes the number of users in ML-LBP, and \( \sigma \) denotes the total number of communities in \{\( C_1, C_2, C_3, C_4 \)\}.

Each step in MLLI can be easily parallelized to reduce its complexity. Specifically, in the step of community division, the graphs in \{\( G_1, G_2, G_3, G_4 \)\} are independent, so we can deploy the Louvain method for graphs in parallel and thus reduce the computation time 75%. As in parallel MP-BP [30, 31], we can also parallelize MP-LBP by calculating the following equation at the beginning of each iteration:

\[
\mathbf{m}^{(t-1)}_{i,z}(x_i(k)) = \prod_{u \in \Gamma_i} \mathbf{m}^{(t-1)}_{u,i,z}(x_i(k)).
\]

Theorem 1: The sufficient and necessary condition that the linearized MP-LBP converges is \( \mathcal{J} < (1/\rho(\mathbf{M})) \). ML-LBP is more complicated and involves many calls of MP-LBP. We could not guarantee whether ML-LBP will converge. When applying ML-LBP in practice, we terminate it empirically until there is no more significant change in the location beliefs in two consecutive iterations, or a maximum tolerable number of iterations are reached.

VII. EVALUATIONS

In this section, we thoroughly evaluate our MLLI scheme using real datasets. We implement all our functions using Python 2.7. All the experiments are carried out on Amazon EC2, with instance type r4.2xlarge, 64GB memory, a 64GiB SSD hard disk, and Ubuntu 16.04 OS.

A. Dataset and Methodology

To evaluate MLLI’s efficacy for datasets with different sizes, we generate three datasets from the whole dataset we collected in Section III. Recall that we have crawled three-month data from December 24, 2016 to March 24, 2017. The three datasets \( S_1, S_2, \) and \( S_3 \) start from December 24, 2016, January 24, 2017, and February 24, 2017, respectively, but all end on March 24, 2017. The number of users and transaction records of \( S_1, S_2, \) and \( S_3 \) are (2.12M, 20.23M), (2.07M, 14.61M), and (1.96M, 6.88M), respectively. Among the 1000 ground-truth users, we select 500 ground-truth users who appear in all three datasets as seeds.

We use the three datasets \( S_1, S_2, \) and \( S_3 \) including the remaining 500 ground-truth users therein to evaluate MLLI. Since it is infeasible to verify the correctness of the inferences result other than the ground-truth users, we evaluate MLLI using the non-seed users of \( S_1, S_2, \) and \( S_3 \). For each of \( S_1, S_2, \) and \( S_3 \), we denote by \( U_1, U_2, \) and \( U_3 \) the 100 non-seed users with most transaction records, the 100 non-seed users with fewest transaction records, and the 100 randomly chosen non-seed users not in \( U_1 \) or \( U_2 \), respectively.

We use sum-based edge weights for category1, category3, and category4 graphs but entropy-based edge weights for
category2 graphs. The reason is that the keywords in category2 correspond to the activities that may happen in the same city or not. If the transactions of some keywords happen consistently, the transaction parties are very likely to be in the same city. Entropy-based edge weights can provide better distinction in such cases and are thus better suited for category2.

B. Inference Accuracy

We first evaluate the accuracy of MLLI. For this experiment, we execute both MP-LBP and ML-LBP for no more than five times, both of which have converged. Let \#TP denote the number of users with correct inferred locations, \#FP denote the number of users with incorrect inferred locations, and \#UN denote the number of users with uncertain locations. For each set \( S_i \) \((i = 1, 2, 3)\) and each set \( U_j \) \((j = 1, 2, 3)\), we calculate \#TP, \#FP, and \#UN as follows. For each user in \( U_i \), if he/she does not belong to the 500 remaining ground-truth users, then \#UN is increased by one. If the user belongs to \( U_i \), if one of the remaining 500 ground-truth users, then the inferred location is the same as one in his/her profile, the value of \#TP is increased by one. Otherwise, \#FP is increased by one.

Table III shows the results for each dataset. The inference accuracy for \( S_i \) and \( U_j \) is defined as

\[
\text{Accuracy}_i = \frac{\#TP_i}{\#TP_i + \#FP_i}.
\]

We consider the top-\(k\) possible locations for each nonseed user. A hit happens as long as his/her true location is in the top-\(k\) possible locations output by MLLI. Fig. 3 depicts the inference accuracy for each dataset. As we can see, MLLI can achieve 83.9\%, 81.5\%, and 82.71\% inference accuracy for \( U_1 \), \( U_2 \), and \( U_3 \), respectively. Since the users in \( U_1 \) have more transaction records than those in \( U_2 \) and \( U_3 \), it is reasonable to see that \( \text{Accuracy}_1 \) is the highest. Likewise, \( \text{Accuracy}_2 \) is of no surprise to be the lowest.

C. Impact of Key Inference Parameters

Now, we evaluate the impact of \( t \), \( l \), and \( \kappa \) on MLLI, where \( t \) and \( l \) denote the maximum number of iterations for MP-LBP and ML-LBP, respectively.

The first experiment checks the relationship between \( t \) and the convergence of MP-LBP. We count the total number of communities in the \( z \)-th category \((\forall z \in [1, 4])\), as well as the number of communities that converge within \( t \) iterations under MP-LBP. Then we define \( \eta_t \) as the later count divided by the former. For each iteration, we utilize the average \( \eta_t = \frac{\sum_z \eta_t}{4} \) to evaluate the convergence of MP-LBP. From Fig. 4(a), we can see that more than 98\% of the communities reach convergence within five iterations under MP-LBP, which indicates the validity of our previous results where \( t = 5 \).

The second experiment checks the relationship between \( l \) and ML-LBP with fixed \( t = 5 \). We define the average probability difference of each user \( i \) in iterations \( l \) and \( l - 1 \) as

\[
\xi^t_{(l)} = \frac{1}{|U_i|} \sum_{j=1}^{|U_i|} \chi^t_{(l)}.
\]

Therefore, the average probability difference of all users is \( \xi^t_{(l)} = \left( \sum_{i=1}^{|U_i|} \chi^t_{(l)} \right) / |U_i| \). The results in Fig. 4(b) show that \( \xi^t_{(l)} \) is smaller than \( 10^{-3} \) when \( l = 5 \), which satisfies our convergence condition. Furthermore, we explore the impact of \( l \) on inference accuracy by varying \( l \) from 1 to 5. Fig. 4(c) shows that: 1) the larger \( l \), the higher inference accuracy, which is as expected and 2) the inference accuracy becomes stable when \( l \) exceeds 5, as our scheme has converged after at most 5 iterations.

Finally, we show in Fig. 4(d) that inference accuracy increases with \( \kappa \). This result is very intuitive.

VIII. RELATED WORK

This paper is most relevant to the prior work on location profiling in OSNs, which can be classified into network-based [4]–[10] and content-based schemes [11]–[15]. Backstrom et al. [4] proposed an estimation algorithm to profile a user’s location, which outperforms IP-based geolocation. Kong et al. [6] extended Backstrom et al.’s work [4] by assigning different weights for neighbors who have potentials to be most predictive of locations. McGee et al. [7] developed a location estimator to predict a user’s location based on the distance between two users and the strength of online social ties. Jurgens [8] proposed to propagate location assignments through social networks with a small number of initial locations. Rout et al. [9] proposed a geolocation inference scheme to infer a users location based on the locations...
of his friends. Jungens et al. [10] systematically compared these schemes [4]–[9] in real-world conditions. All these schemes [4]–[9] assume that the probability of being online friends for given physical distance is the same for different users. Obviously, this assumption may not hold in practice; e.g., a famous user is more likely to have a follower far away than a regular user. In addition, Cheng et al. [11] predicted a user’s location based on the content of tweets, where they identified a set of geographic hints (e.g., “New York”) and utilized them to associate the user with some locations. Chandra et al. [12] proposed an improvement over [11] by associating a user’s original tweets to himself and his retweets to the initial users. Furthermore, Mahmud et al. [13], [14] utilized an ensemble of statistical and heuristic classifiers to predict locations, but these schemes treat geographic hints and locations as discrete labels and overlook their explicit relations. Later on, Li et al. [32] explored a probabilistic model to profile users’ locations, which utilizes OSN connections and tweets in a unified and discriminative manner. Finally, Zhang et al. [33] proposed a novel and lightweight system to find the majority of the users in a specific geographical area without scanning the whole Twittersphere. Public transaction records in Venmo do not have sufficient geographic cues. So this technique [33] is not directly applicable to our context.

There is also significant research on inferring sensitive user information other than locations. For example, Zhang et al. [34] proposed a new framework to profile the hidden ages of microbloggers by exploring public content and interaction information. Nguyen et al. [35] explored the relation between age and language use for inferring age categories, life stages, and exact ages. Besides, Weisberg et al. [36] proposed to infer gender by considering the rating scores for different movies. Jia et al. [25] explored user behaviors to learn a binary classifier and then used it to predict the prior probability that each target user has a specific attribute value. Then they used a binary random variable to characterize each user and modeled the joint probability distribution of all binary random variables as a pMRF based on the OSN structure. Given the training dataset and prior probability, they used loopy belief propagation to propagate label information and obtained the posterior probability to predict whether a target user has the attribute or not. This work [25] motivates us to use a pMRF for MP-LBP.

IX. Conclusion

In this paper, we studied the implicit leakage of personal location information in Venmo, a popular mobile P2P system. We developed MLLI, a novel MLLI algorithm to infer hidden user locations from public transaction records in Venmo. Based on a real dataset consisting of 2.12M users and 20.23M transaction records, we showed that MLLI can identify the top-3 possible locations for a Venmo user with accuracy up to 80%.

Our experimental results show that the attackers can infer Venmo users’ home locations with high accuracy. Here, we briefly discuss two potential countermeasures. First, Venmo App may change its default system setting for users’ transactions. In particular, the current default system setting is that user’s transactions are public, and we recommend to change it to nonpublic. As reported by the Rainie and Duggan’s reports [37], most people are not aware of potential privacy risks, even giving up privacy for convenience [38]. Since users seldom change system default settings, doing so can effectively prevent the attacker from learning the temporal patterns of user transactions and thus thwart location inference. Second, Venmo APP can reduce the number of public transaction records and thus dilute the weights among users. This can decrease the inference accuracy as the belief propagation processes in the MLBP technique mainly depend on the weights among users.

REFERENCES


Yimin Chen (S’14) received the B.S. degree in electrical engineering from Peking University, Beijing, China, in 2010, and the M.Phil. degree in electrical engineering from the Chinese University of Hong Kong, Hong Kong, in 2013. He is currently pursuing the Ph.D. degree at the School of Electrical, Computer, and Energy Engineering, Arizona State University, Tempe, AZ, USA.

His current research interests include security and privacy issues in computer and networked systems.

Yanchao Zhang (S’03–M’06–SM’11) received the B.E. degree in computer science and technology from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 1999, the M.E. degree in computer science and technology from the Beijing University of Posts and Telecommunications, Beijing, China, in 2002, and the Ph.D. degree in electrical and computer engineering from the University of Florida, Gainesville, FL, USA, in 2006.

He is a Professor with the School of Electrical, Computer, and Energy Engineering, Arizona State University, Tempe, AZ, USA. His current research interests include security and privacy issues in computer and networked systems, with current focus areas in emerging wireless networks, mobile crowdsourcing, Internet of Things, social networking and computing, wireless/mobile systems for disabled people, big data analytics, mobile/wearable devices, and wireless/mobile health.

Dr. Zhang was a recipient of the U.S. NSF CAREER Award in 2009. He has been on the Editorial Boards of the IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE WIRELESS COMMUNICATIONS, the IEEE TRANSACTIONS ON CONTROL OF NETWORK SYSTEMS, and the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY. He also chaired the 2017 IEEE Conference on Communications and Network Security, the 2016 ARO Workshop on Trustworthy Human-Centric Social Networking, the 2015 NSF Workshop on Wireless Security, and the 2010 IEEE GLOBECOM Communication and Information System Security Symposium.

Yaping Lin (M’15) received the B.S. degree in computer application from Hunan University, Changsha, China, in 1982, the M.S. degree in computer applications from the National University of Defense Technology, Changsha, in 1985, and the Ph.D. degree in control theory and application from Hunan University, in 2000.

He has been a Professor and a Ph.D. Supervisor with Hunan University since 1996. From 2004 to 2005, he was a Visiting Researcher with the University of Texas at Arlington, Arlington, TX, USA. His current research interests include machine learning, network security, and wireless sensor networks.