Hyperlocal: Inferring Location of IP Addresses in Real-time Bid Requests for Mobile Ads

Long T. Le & Tina Eliassi-Rad
Rutgers University
{longtle, eliassi}@cs.rutgers.edu

Foster Provost
New York University
fprovost@stern.nyu.edu

Lauren Moores
Dstillery
lmoores@dstillery.com

Abstract

To conduct a successful targeting campaign in mobile advertisement, one needs to have reliable location information from real-time bid requests. Due to privacy and other reasons, many of real-time bid requests do not include latitude and longitude information. In this paper, we present a three step approach that takes in real-time bid requests and (1) creates a massive weighted heterogenous network, (2) applies network-inference techniques to infer fine-grain (but possibly noisy) location information for *location candidates* represented by hashed public IP addresses, and (3) uses k-nearest neighbor and census data to assign census block group IDs to location candidates (i.e., hashed public IP addresses in real-time bids). The accuracy of our approach is over 74% for hashed IPs (regardless of their type: mobile or non-mobile). This is notable since our inference is over 212K possibilities.

1 Introduction

When targeting mobile advertisements, having reliable location information for real-time bid requests is an important part of conducting a successful campaign. However, many real-time bid (RTB) requests do not include fine-grained location information (such as latitude and longitude). For example, in a recent collection of approximately 44 million requests from six different Supply Side Platforms (SSPs), 37% of the RTB requests had no location information. Based on the traffic type, the percentage of requests without location information can be as high as 84%.[1] For privacy reasons, some SSPs provide no location information at all. Furthermore, due to IP address translation, targeters cannot simply look up fine-grained location of IP addresses in common databases. There are two reasons for this: (1) the RTB systems simply do not have this information; and (2) there is a possible privacy concern with processing and storing fine-grained location information.

In this paper, we study a fast and scalable approach to solving the problem of inferring locations of IP addresses in real-time, mobile bid requests at the Census Block Group (CBG) level. Our working assumption is that CBGs comprise location information fine-grained enough for useful hyper-local ad targeting, yet coarse-grained enough to avoid major privacy concerns. (A CBG typically covers a contiguous area and contains between 600 and 3,000 people; the United States is divided into approximately 212K CBGs.)

Our proposed approach has three steps. First, we create a weighted heterogeneous "movement network" among the location candidates. The nodes in this network are represented by hashed IP addresses. Novelties of our movement network are: (a) the separation of mobile and non-mobile IP addresses; and (b) the storing of inter-arrival times (IATs) and number of movements on each edge of the network. Second, we apply local relational classifiers with movement- and IAT-based weights to infer the fine-grained location of the candidates without location information. This fine-grained information can be noisy. Third, we assign CBG IDs to the predicted (and possibly noisy) fine-grained location information by using a new procedure that employes k-nearest neighbor and census data. We conduct an extensive empirical study. Our accuracy on matching CBG ID is over 74% for all hashed IP types (mobile and non-mobile). This is notable since we are estimating the correct CBG out of approximately 212K possibilities.

Our main contributions are as follows:

- Given RTB requests, we build a massive weighted heterogeneous movement network among the location candidates, represented by hashed public IP addresses.
- We employ local network-inference techniques with movement- and IAT-based weights to predict latitude and longitude values for candidates without location information.
- We introduce a k-nearest neighbor procedure to assign CBG IDs to predicted (and possibly noisy) latitude and longitude information.

[1] Traffic from mobile applications tend to have more location information.

Figure 1: The eco-system of mobile real-time bidding ads. Our work is on data from supply-side platforms (a.k.a. supply-side provides).

- We demonstrate the effectiveness of our approach on a massive real-world data set and observe an accuracy of over 74% for all hashed IP types (mobile and non-mobile). We observe that accuracy on hashed non-mobile IPs is over 55% (a harder task described in detail later). However since we are inferring the correct CBG out of approximately 212K possibilities, our results are noteworthy.

The paper is organized as follows. Next we discuss background and related words. In Section 3, we introduce our proposed method. Section 4 presents our experiments and discussion. We conclude the paper in Section 5.

2 Background and Related Work

Figure 1 depicts the mobile real-time bidding ad eco-system.[2] Accurately inferring the location of an IP address is important in many application. To our knowledge, our work is the first of its kind that uses just the structure of an IP×IP movement graph to infer locations, in terms of Census Block Group IDs, for hashed public IP addresses.

Wong et al. [11] propose a framework for locating IPs by representing node positions through regions, expressing constraints as areas, and computing locations by solving a system of geometric constraints. In another study, Wang et al. [10] develop a client-independent geolocation system. Both of these methods rely on pings to estimate the round trip time (RTT) between two machines. They also rely on landmarks, which are collected manually. Our approach does not have these limitations.

Balakrishnan et al. [1] study geolocating IP addresses on mobile networks. They examined the properties of cell-phone IP addresses. Their study showed that mobile IPs are ephemeral and their addresses are itinerant. For example, an individual cell phone can report different IP addresses to various servers within a short time-period. This makes it very difficult to track the device without having any software on them (which is our case).

The same public IP address is often used by many devices. Metwally et al. [7] estimate the number of users of an IP address by keeping track of the application-specific traffic, which can be costly.

There are previous studies that use social interaction to model user movement and predict the future location of a user [2], [8], [3]. These methods require information about the social relation between users through phone calls or friendship on an online social network. A finding of these studies was that long distance movements are influenced by ties in the social network. In our work, we do not have access to such information.

[2]Figure from `http://businessinsider.com`.

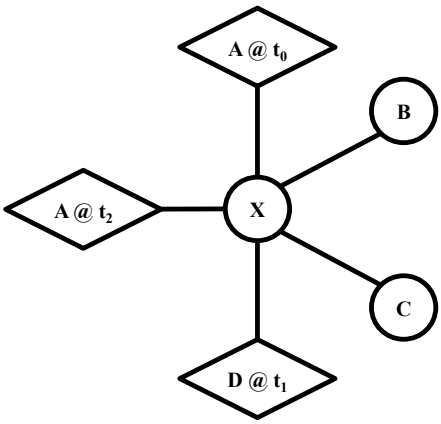


Figure 2: A small sample of an IP×IP movement graph. The circle nodes are non-mobile IPs. The diamond nodes are mobile IPs. The mobile IPs are time-stamped because they are transient. To not clutter the picture, we do not show the weights on edges, which are the number movements between two nodes and the inter-arrival times for all movements on an edge.

3 Proposed Method

Before we start this section, it is important to note that the IP address which one finds when one looks up the address from one's computer (e.g. via `ipconfig`) is **not** the IP address that is seen in real-time bidding (RTB) systems. An address translation is done between the two, which results in the public IP address. A public IP, provided by the Internet Service Provider, is the address that the outside world sees (e.g., a Web server sees when one logs into its Web site). These public IP addresses are then hashed.

Our proposed method has three steps: building IP×IP movement graph, employing local relational classifiers, assigning census block groups as proxies for location.

3.1 Building IP×IP Movement Graph

Given that we are interested in inferring the location of an IP address, we represent the RTB-request data as a network of movements between heterogenous IP nodes.[3] In particular, we have two different types of IP addresses: mobile and non-mobile. Examples of mobile IP addresses are ones with connection types 3G, 4G, LTE, etc. Examples of non-mobile IP addresses are ones with connection types broadband, xDSL, cable, etc. Figure 2 shows a small sample of an IP×IP movement graph.

Figure 3 illustrates how we determine the type of an IP address. We initially relied on a 3rd party database to classify mobile and non-mobile nodes. However, we noticed that a non-negligible percentage of the IPs classified as non-mobile by the 3rd party database appeared outside of a radius r in 24 hours, which is a characteristic trait of a mobile IP (see Section 4). So, we added an extra condition on whether an IP is non-mobile (i.e., did the IP appear outside radius r in 24 hours?). We attempted various radius values in our experiments (See Section 4). However, the default setting for the radius r is $100m$ because of the following reasoning. Non-mobile devices use Wi-Fi routers to access the Internet. The range of Wi-Fi routers depends on the routers' antenna technology and surrounding condition. The range of current routers is normally less than 100m.[4]

Two nodes, IP_A and IP_B, have an edge if the same network exchange identifier[5] (a.k.a. NUID) uses IP_A at time t_1 and then uses IP_B at time t_2. Since we are using IP addresses as proxies for locations, a mobile IP address IP_A at time t_i is represented by a node $\langle IP_A, t_i \rangle$ and a non-mobile IP address IP_B at time t_j is represented by IP_B only. This asymmetry is due to mobile IP addresses being more transient than non-mobile IP addresses.

The inter-arrival time(s) of an NUID's movement from IP_A at t_1 to IP_B at t_2 is stored on the edge. We distinguish between three different types of movements: (1) movement between two non-mobile IP addresses, (2) movement

[3] We use the terms 'network' and 'graph' interchangeably in this paper.

[4] `http://compnetworking.about.com/cs/wirelessproducts/f/wifirange.htm`

[5] A network exchange identifier is associated with each device. This information is not always provided.

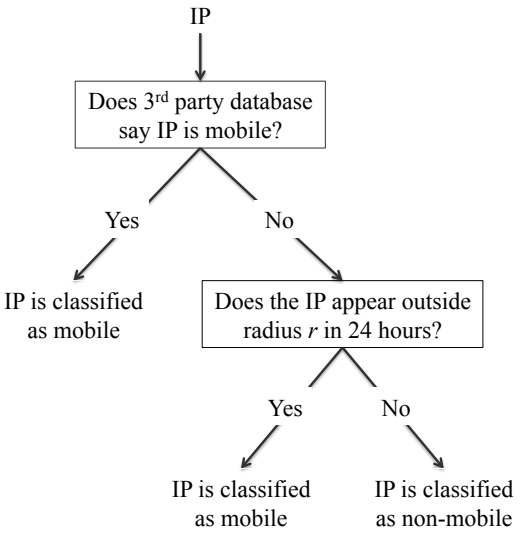


Figure 3: Decision Procedure for determining if an IP address is mobile or non-mobile. The extra condition on whether an IP is non-mobile is added because mobile IP addresses tend to change position more quickly than non-mobile IP addresses [1].

between a non-mobile and a mobile IP addresses, and (3) movement between two mobile IP addresses. In the last case, two mobile IPs are represented by a tuple $\langle IP_A, t_1 \rangle$ and $\langle IP_B, t_2 \rangle$; and, we only consider this a movement if IP_A is different from IP_B. We stress this condition since there are cases where one NUID is involved in two consecutive RTB requests from the same hashed mobile IP address. For example, at time t_1 we observe an RTB request involving $NUID_1$ at hashed mobile IP address IP_A and this same $NUID_1$ is observed again in another RTB request at time t_2 with the same hashed IP address IP_A. Due to the way we represent the mobile IP addresses, this NUID appears at $\langle IP_A, t_1 \rangle$ and $\langle IP_A, t_2 \rangle$, but we do not consider this is a movement and do not add an edge between them.

There may be multiple movements between two nodes in our IP $\times$ IP movement graph. For each edge in our movement graph, we keep track of the number of movements and inter-arrival times (IATs) between them. Number of movements is the number of times that we observe one NUID use the first IP and then change to using the second IP. The IAT of a movement is the time gap between the appearances of NUID when it changes from one IP address to another. Since there are multiple movements between IPs, we actually have distributions of IATs over the edges. However, in our experiments, we only use the minimum IAT of all the movements on an edge since a smaller IAT indicates a smaller distance.

3.2 Employing Local Relational Classifiers

When inferring location on the IP$\times$ IP movement graph, it is desirable to do inference locally. This is because (1) the farther out one moves in this movement graph, the farther away one gets geographically; and (2) the movement graph is often very large so non-local approaches can be computationally burdensome.

For our local relational classifier, we utilize *wvRN*, which stands for *weighted-vote Relational Neighbor* classifier [6]. wvRN estimates class membership probabilities using the assumption of *homophily* (i.e., like attracts like) in the network data. Given the existence of homophily, wvRN performs well when compared to more complex classifiers [6].

The key for our inference problem is what weight to use in wvRN. We consider two weights: number of movements and minimum IAT. We refer to the former as `wvRN(numMoves)` and the latter as `wvRN(minIAT)`.

`wvRN(numMoves)` uses the number movements between two IP addresses as the weight in wvRN. The intuition behind this weight is that a node v will be closer in distance to its neighbors with whom it has more movements. So, w_i is the number of movements between v and its i-th neighbor. Our data, like many other real-world data sets, is very

sparse – i.e., it has many edges with only one movement. This inspired us to try another weight based on IATs.

wvRN(minIAT) uses the normalized minimum inter-arrival times (IAT) between two IP addresses. It makes sense for IAT to be a good indicator for the distance between two nodes since the longer the IAT, the longer distance the user has potentially moved (sans traffic). We use $\texttt{minIAT}_v$ to denote the minimum IAT on all edges of a node v. Then, the weight on the movement between v and its neighbor i is defined as

$$w_i = \frac{\texttt{minIAT}_v}{\min(t),\ \forall t \in IAT(v,i)}$$

where $IAT(v,i)$ returns the list of IATs between v and i. Note that in the above formula, the denominator is never strictly smaller than the numerator. So, the largest value for w_i is 1 indicating that there was a movement between v and its neighbor i that took $\texttt{minIAT}_v$ time.

wvRN Equations: Given a node v, its neighbors $Nbr(v)$ who have location information, and the weights on the edges between v and its neighbors (where w_i is the weight on the edge from v to its i-th neighbor), we use the following equations to predict latitude and longitude values for v:

$$latitude(v) = \frac{\sum_{i \in Nbr(v)} w_i \times latitude(i)}{\sum_{i \in Nbr(v)} w_i}$$

$$longitude(v) = \frac{\sum_{i \in Nbr(v)} w_i \times longitude(i)}{\sum_{i \in Nbr(v)} w_i}$$

Restricting Inter-arrival Times on IPs with One Known Neighbor: Like most real-world graphs, the IP $\times$ IP movement graph has a skewed degree distribution (see Figure 7), with many nodes having only one neighbor. By putting a constraint on the IAT of IPs with only one neighbor (e.g., that the IAT has to be less than or equal to 60 min), we can effectively prune the noisy links from the IP $\times$ IP movement graph. The only issue here is that with pruning, one also reduces the size of the inference set. We explore this issue in-depth in Section 4.

3.3 Assigning Census Block Groups as Proxies for Location

We infer the location of a hashed public IP address at the *Census Block Group* (CBG) level instead of the $\langle latitude, longitude \rangle$ level. The US Census Bureau defines a census block as the smallest geographic unit used in tabulation of data collected from all residences. The US (including Puerto Rico) has about 8.2M census blocks.[6] As the name suggests, a CBG is a group of census blocks, which are close geographically and never cross state or county boundaries. The US (including Puerto Rico) has about 212K CBGs, each containing an average of 39 blocks and between 600 and 3000 people.[7] We decided to infer location at the CBG level because: (1) it provides a more consistent labeling for location of IP addresses; (2) it allows incorporation of external data that uses census data such as demographics; and (3) in the majority of mobile applications (e.g., mobile ads), this level of location information is sufficient for a successful campaign.

Given the predicted $\langle latitude, longitude \rangle$ of an IP address from wvRN (see previous section), we need a method for assigning a CBG ID to it. Our procedure, a k-nearest neighbor approach, is as follows:

Inputs:

- Location of interest, $loc = \langle lat, lon \rangle$
- For each CBG i in the US, i's centroid $c_i = \langle lat_i, lon_i \rangle$ and i's area a_i in km

Output:

- The CBG ID that contains loc

[6] http://en.wikipedia.org/wiki/Census_block

[7] These statistics are from the 2000 census. For more details on CBGs, see http://www.census.gov/.

Data Name	Collection Date	# of RTB Requests with US Valid NUIDs	% RTB Requests **without** Location	% RTB Requests from **Mobile** IPs
Oct-2012	$10/01/2012$	$44,137,669$	36.5%	57.3%
Feb-2013	$02/06/2013$	$21,551,483$	56.7%	47.7%

Table 1: Some characteristics of our two datasets: one collected on Monday 10/01/2012 and the other on Wednesday 02/06/2013. The number of real-time bid requests decreased by about 51% from October 2012 to February 2013 due to reductions from some supply side providers. However, the number of requests without location information increased by about 55%. The number of requests from mobile IPs decreased by about 17%. We collected data from another day in February 2013 and the characteristics were similar to 02/06/2013.

Procedure:

1 $C \leftarrow$ centroids of the k nearest CBGs to loc

2 For j in C

$d_j \leftarrow distance(loc, c_j)$ // distance between loc and the centroid of the j-th nearest CBG

$r_j \leftarrow \sqrt{(\frac{a_j}{\pi})}$ // CBG radius of the j-th centroid

$ratio_j \leftarrow \frac{d_j}{r_j}$ // ratio of distance to radius

3 Return the CBG ID corresponding to $min(ratio_j), \forall j \in C$

Since we compute the latitude and longitude of an IP based on neighboring IP addresses' latitudes and longitudes, a location may be returned that is in the middle of a lake, forest, or desert. In such cases, the minimum ratio is high; and it is unreasonable to return a CBG ID. Thus, our algorithm will return "unknown" in such cases. Specifically, if the minimum ratio $(\frac{d_j}{r_j})$ is over a threshold t, our procedure returns "unknown" for the CBG ID of the given IP. In our experiments, the percentage of CBG IDs returned as "unknown" was less than 1% with $k = 5$ and $t = 2$. See Section 4.

4 Experiments

This section is organized as follows: data description, experimental setup, results, and discussion.

4.1 Data Description

We conducted experiments on two real-world data sets. Table 1 lists the basic characteristics of each data set. For the experiments, we only consider RTB requests with valid USA NUIDs. Recall that NUID is the network exchange identifier. Each device has an NUID; but it is not always provided by the RTB system. Also, NUIDs for the same device may be different across different SSPs.

Figure 4 provides details about the data sets such as distribution of RTB requests over SSPs, and the conditional distributions of RTB requests over SSPs given only the requests with location, and then given only the requests without location information. Recall that we may not have location information for a request because (1) the RTB system did not forward it, (2) the SSP did not forward it, (3) the device did not capture it, or (4) the user did not enable location-based services.

Figure 5 shows the distribution of requests with and without location information per SSP. All requests from some SSPs – e.g., SSP1 and SSP3 – are without location information. On the other hand, some SSPs – e.g., SSP7 – provide location information for all of their requests.

Table 2 reports the amount of *homophily* (i.e., like attracting like) in our IP $\times$ IP movement graphs. We define homophily as the number of movements whose (IP) endpoints are from the same SSP divided by the total number of all movements. Homophily is high (greater than 90%) in both data sets. The Feb-2013 data has less homophily

Homophily	Oct-2012	Feb-2013
All movements	96.5%	90.8%
Mobile to mobile movements	98.6%	99.2%
Non-mobile to non-mobile movements	86.9%	78.1%

Table 2: Homophily in IP $\times$ IP movement graphs. Homophily is defined as the number of movements whose (IP) endpoints are from the same SSP divided by the total number of all movements. Homophily levels are high in both data sets. Movements between mobile IPs have very high homophily.

(by $\approx 5.7\%$) than the Oct-2012 data. Homophily between mobile IPs is very high (over 98%). Homophily between non-mobile IPs is lower than that of mobile IPs (namely, 86.9% for Oct-2012 and 78.1% for Feb-2013).

Given the high levels of homophily in the IP $\times$ IP movements graphs, can we predict location for IPs whose requests are from SSPs without location information? The answer to this question is yes. We observe sufficient non-homophily in the graphs that if an SSP does not have location information for its requests, we can still predict location for its IPs (because its IPs are likely to be linked to other IPs from SSPs with location information). Figure 6 depicts this non-homophily for SSP1 and SSP3, which do not provide any location information for their requests. More than 50% of IPs from these SSPs have movements to IPs from other SSPs (which provide location information).

4.2 Experimental Setup

We show results for two different methods: `wvRN(minIAT)` and `wvRN(numMoves)`. See Section 3 for details on these methods. Recall that we measure accuracy by checking the predicted CBG ID vs. the actual CBG ID of an IP address.

We divide our experiments into nine combinations of: *infer location for X using Y*. Values for X are 'all IPs', 'mobile IPs', and 'non-mobile IPs'. Values for Y are 'all neighbors', 'mobile neighbors', and 'non-mobile neighbors'. As discussed before, these IPs are all public IP addresses that have been hashed.

Figure 7 shows the degree distribution for these nine combinations with all neighbors and with only known neighbors (i.e. neighbors with location information) for the Oct-2012 data. The plots for the Feb-2013 data are similar and were omitted for brevity. We observe that except for degree distribution over mobile IPs, the rest follow power-law distributions with heavy tails. The mobile degree distributions are different because we represent them as time-stamped nodes (as described in Section 3.1).

Also, the distributions for all neighbors and known neighbors are very similar. We do not observe sparsity around the neighbors an IP. For example, in the Oct-2012 data approximately 70% of an IP's neighbors are known. Therefore, we do not attempt more computationally expensive relational-learning methods like collective classification [9, 6, 5, 4].

4.3 Results

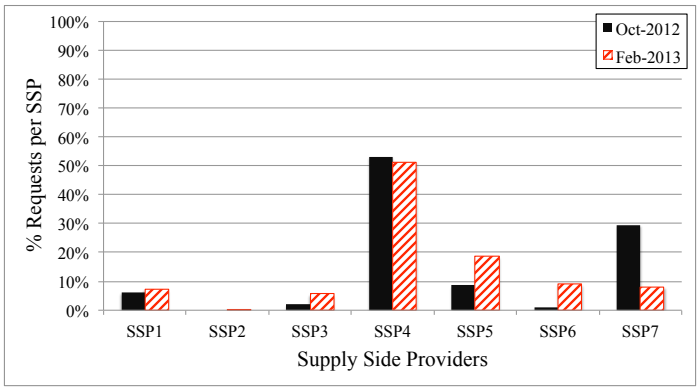
We ran our experiments on a Macbook Pro with CPU 2.66 GHz Intel Core i7, RAM 8 GB DDR3, hard drive 500 GB SSD, and OS X 10.8. Our algorithm is implemented in Python. We use NetworkX[8] and MongoDB[9] for network management and for finding k-nearest neighbors. It takes us on average 1.2 milliseconds to process each bid request.

Our results are organized as follows: (1) sensitivity to radius r, (2) sensitivity to inter-arrival times, (3) core results, (4), inference over IPs with one known neighbor, (5) inference over IPs with two or more known neighbors, and (6) accuracy with a slack distance. Again, we would like to stress that all IPs are hashed public addresses.

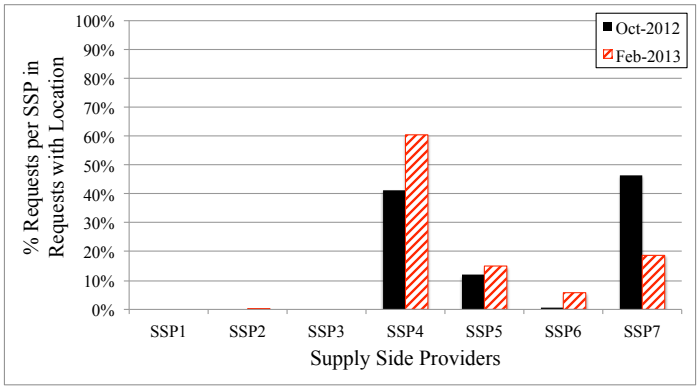
4.3.1 Sensitivity to Radius r

Previously, we illustrated our procedure for deciding whether an IP address is mobile or non-mobile (see Figure 3). We had a radius parameter r, which checked if an IP appeared outside the radius r in 24 hours. What percentage of the IPs do we reverse from the 3rd party decision (i.e., reclassify a non-mobile IP to a mobile IP because it appeared outside the radius r in 24 hours)? Figure 8 answers this question. At the default radius of 0.1 km, we respectively reverse 10.5% and 12% of the decisions made by the 3rd party for Oct-2012 and Feb-2013.

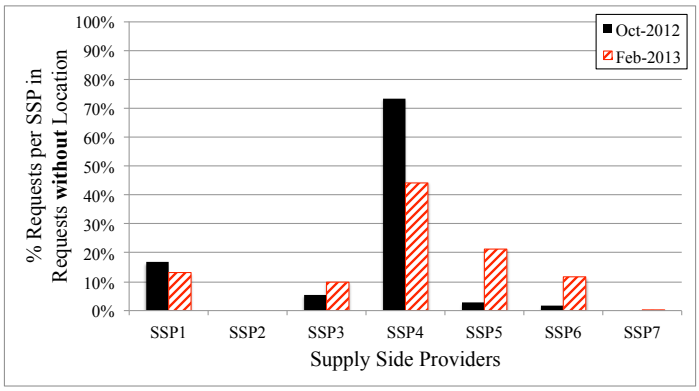
[8] `http://networkx.github.io`
[9] `http://docs.mongodb.org`



(a) % Requests per SSP



(b) % Requests per SSP in the requests with location information



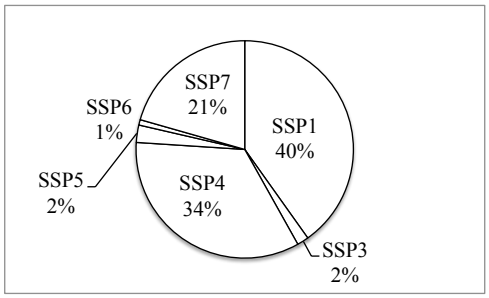
(c) % Requests per SSP in the requests **without** location information

Figure 4: Data characteristics per supply side provider (SSP). (a) SSP4 provides about 50% of the requests for both datasets. (b) Among the requests with location information, SSP4 and SSP7 dominate. (c) Among the requests without location information, SSP4 dominates.

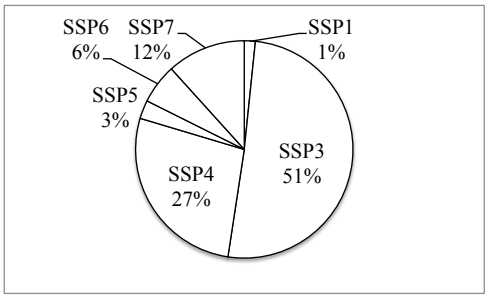
(a) Oct-2012

(b) Feb-2013

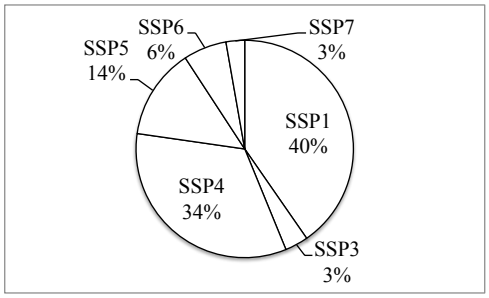
Figure 5: Percentage of requests with and without location information per SSP. None of the requests from SSP1 and SSP3 have location information. However, all the requests from SSP7 have location information.



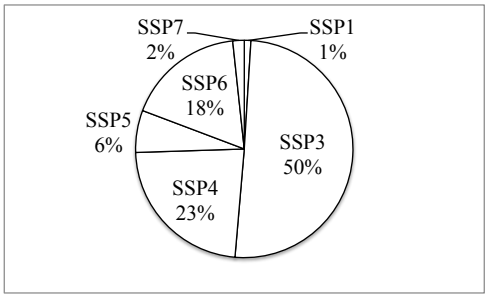
(a) Oct-2012: Movements between SSP1 & other SSPs



(b) Oct-2012: Movements between SSP3 & other SSPs

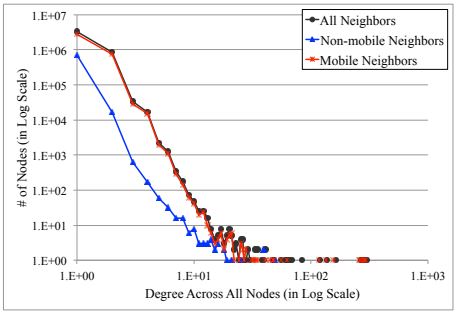


(c) Feb-2013: Movements between SSP1 & other SSPs

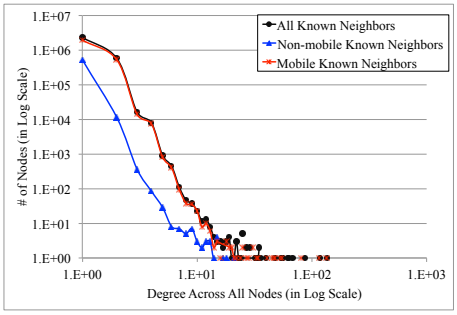


(d) Feb-2013: Movements between SSP3 other SSPs

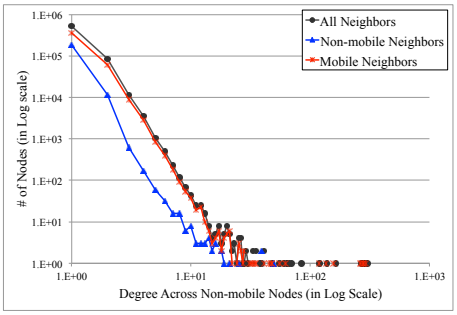
Figure 6: Percentage of IPs from SSP1 requests and from SSP3 requests that connect to other SSPs. We see that there is a considerable overlap with other SSPs that provide location information.



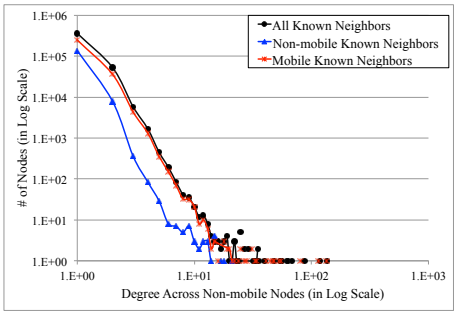
(a) Degree distribution over all nodes considering known & unknown neighbors



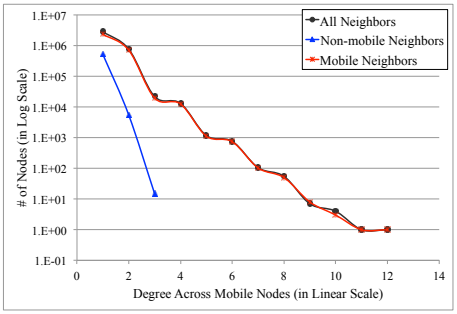
(d) Degree distribution over all nodes considering only known neighbors



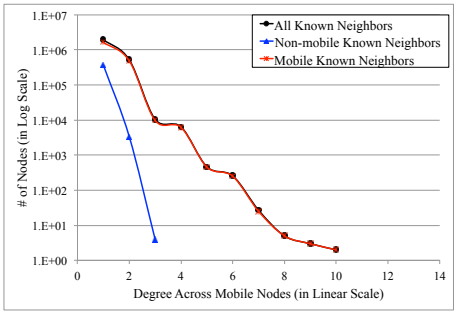
(b) Degree distribution over non-mobile nodes considering known & unknown neighbors



(e) Degree distribution over non-mobile nodes considering only known neighbors



(c) Degree distribution over mobile nodes considering known & unknown neighbors



(f) Degree distribution over mobile nodes considering only known neighbors

Figure 7: Oct-2012 degree distributions for the various combinations of inferring location for X given Y. Values for X are all IPs, mobile IPs, and non-mobile IPs. Values for Y are all neighbors, mobile neighbors, and non-mobile neighbors. Given how we represent mobile IPs as time-stamped nodes, we expect a different distribution in (c) and (f) than in other plots. The degree-distributions plots for the Feb-2013 data are similar to these plots and are omitted for brevity.

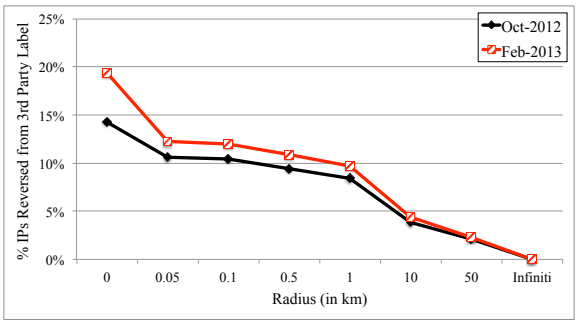
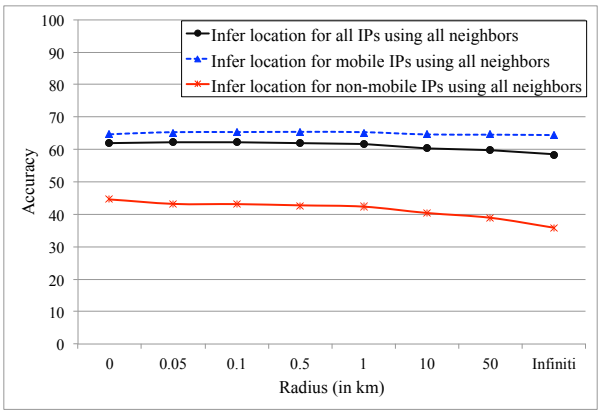
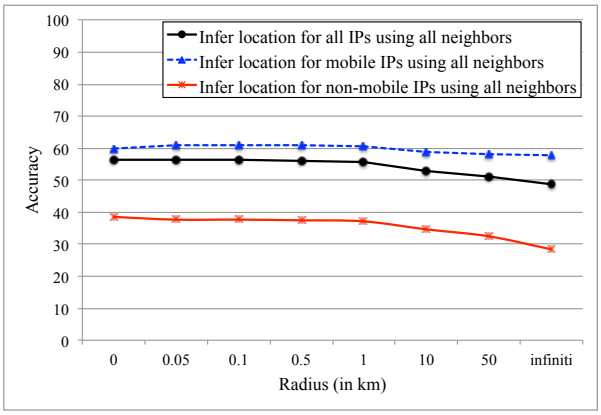


Figure 8: Percentage of the IPs whose classification we reverse from the 3rd party decision (see Figure 3 for the decision procedure). A radius of infiniti means no constraint was assigned. At our default radius of 0.1 km, we reverse 10.5% of the classifications for the Oct-2012 data, and 12% for the Feb-2013 data.



(a) Oct-2012: Accuracy vs. Radius



(b) Feb-2013: Accuracy vs. Radius

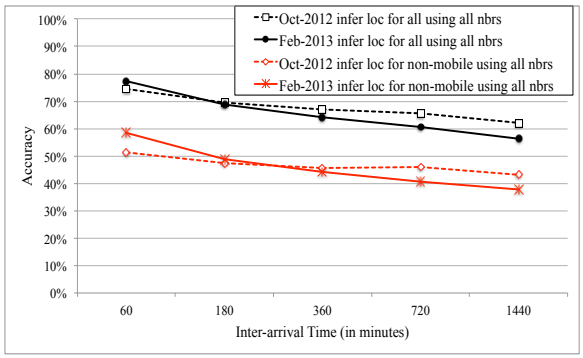
Figure 9: Sensitivity of `wvRN(numMoves)` to various radii in the decision procedure for mobile vs. non-mobile classification (see Figure 3). A radius of infiniti means no constraint was assigned. The trends in the two data sets are the same. In Oct-2012, the maximum difference is an 8% decrease in accuracy between a radius of 0 and a radius of infiniti; it appears when we are inferring location for non-mobile IPs using all neighbors (the red plot). The same holds true for Feb-2013, but the maximum difference is about 10%.

How sensitive are our accuracy results to change in the radius parameter? Figure 9 answers this question. We observe that different radii do not significantly change the accuracy of `wvRN(numMoves)`.[10] From a radius of zero to a radius of infiniti, the change in accuracy is less than 10%. In Figure 9, we only plot results for inferring location on (1) all IPs, (2) only mobile IPs, and (3) only non-mobile IPs. In these three cases, we use all neighbors in the inference. The other combinations of this experiment (e.g., infer location of non-mobile IPs using non-mobile neighbors, infer location of non-mobile IPs using mobile neighbors, etc) show the same pattern, so we have omitted them for brevity. For the rest of the experiments, we use a radius r of 0.1 km (or 100 m). In Section 3.1, we gave a technical reason for why r = 0.1 km is a reasonable choice.

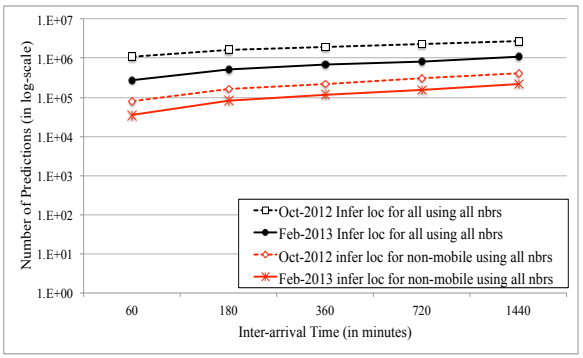
4.3.2 Sensitivity to Inter-arrival Times (IAT)

Figure 10 shows accuracy and number of predictions as we vary IAT. We use `wvRN(numMoves)` for inference. The plots show the results on inferring location for all IPs using all neighbors and for non-mobile IPs using all neighbors. The values for mobile IPs are similar to all IPs and are omitted from the plot. The results for the other combinations of infer on X using Y neighbors are also similar and are omitted for brevity. We observe that as IAT is restricted

[10] In this version of wvRN(numMoves), we did not limit IATs for IPs with one known location.



(a) IAT vs. Accuracy



(b) IAT vs. Number of Predictions

Figure 10: Accuracy and number of predictions across different IATs. `wvRN(numMoves)` is used for these runs. As IAT is restricted to smaller values, the number of predictions has a sharper drop (relatively speaking) than the gain in accuracy. The results on Oct-2012 and Feb-2013 data sets are similar. The Feb-2013 has less predictions because it is a smaller data set. Radius r = 100m.

to smaller values, the number of predictions has a bigger drop (relatively speaking) than the gain in accuracy. These results inspired us to use IATs as weights in `wvRN`; and so we developed the `wvRN(minIAT)` method.

4.3.3 Core Results

Table 4.3.3 reports our core results in terms of (1) accuracy of `wvRN(minIAT)` vs. `wvRN(numMoves)`, (2) number of predictions made (i.e. size of the inference set), and (3) percentage of mobile IPs in predictions. We show results for what type of IP we are inferring location (namely, all IPs vs. mobile IPs vs. non-mobile IPs) and what type of information we are using (i.e., all neighbors vs. mobile neighbors vs. non-mobile neighbors). Our observations are as follows:

- Even though the accuracy of `wvRN(minIAT)` is higher than that of `wvRN(numMoves)`, this difference is not statistically significant at the 0.05 level.
- Both methods have high accuracy (73% to 79%) on inferring location for mobile IPs and all IPs when using mobile neighbors. In the latter case (when inferring for all IPs), between 87.4% to 93.2% of the predictions are on mobile IPs. The range depends on the particular data set.
- The best accuracy for non-mobile IPs is 55.4% in Oct-2012 and 62.7% in Feb-2013. This is when we use non-mobile neighbors.
- It is more difficult to infer the location of a non-mobile IP than a mobile IP. This is because mobile IPs tend to have smaller inter-arrival times and smaller distances than non-mobile IPs. In the Oct-2012 data, 77.1% of the mobile to mobile movements were less than 1 km; while only 44.2% of the non-mobile to non-mobile movements were less than 1 km. These numbers are similar for the Feb-2013 data.
- The number of non-mobile IPs in the data sets are small compared to mobile IPs. See Table 4.3.3. This is an artifact of how we represent the mobile IPs (as unique time-stamped nodes). See Section 3.

4.3.4 Inference over IPs with One Known Neighbor

Like many real-world networks, our data is highly skewed w.r.t. its degree distribution, with many IPs having only one known neighbor (see Figure 7). Tables 4.3.4 and 4.3.4 show `wvRN(numMoves)` results of inference on IPs with only one known neighbor. The former table showcases inference on the neighbors whose IAT is less than or equal to 60 minutes; and the latter table showcases inference on the neighbors whose IAT is greater than 60 minutes. The first observation here is that for IPs with only one known neighbor, restricting IAT to be less than 60 mins improves

Infer location for **all** IPs . . .	Accuracy wvRN(minIAT) Oct-2012 | Feb-2013	Accuracy wvRN(numMoves) Oct-2012 | Feb-2013	Number of Predictions Oct-2012 | Feb-2013	% Mobile in Predictions Oct-2012 | Feb-2013
using all neighbors	71.6 | 70.8	69.7 | 69.3	1,189,679 | 328,015	91.0% | 83.8%
using mobile neighbors	**74.4** | **74.8**	**72.6** | **73.5**	1,077,644 | 278,674	93.2% | 87.4%
using non-mobile neighbors	51.9 | 57.7	51.7 | 57.3	98,338 | 40,777	68.7% | 62.1%
Infer location for **mobile** IPs . . .				
using all neighbors	74.2 | 75.0	72.3 | 73.6	1,082,566 | 274,900	100% | 100%
using mobile neighbors	**76.6** | **79.1**	**74.7** | **77.9**	1,004,601 | 243,630	100% | 100%
using non-mobile neighbors	50.4 | 54.7	50.2 | 54.5	67,553 | 25,323	100% | 100%
Infer location for **non-mobile** IPs . . .				
using all neighbors	45.5 | 49.0	44.0 | 46.8	107,113 | 53,115	0% | 0%
using mobile neighbors	44.0 | 45.1	42.5 | 43.0	73,043 | 35,044	0% | 0%
using non-mobile neighbors	**55.4** | **62.7**	**55.0** | **62.0**	30,785 | 15,454	0% | 0%

Table 3: Core results: wvRN(minIAT) vs. wvRN(numMoves). For each method, the highest accuracy values are in boldface. The differences between the two methods are not statistically significant at the 0.05 level. The number of predictions varies depending on the particular inference and the types of neighbors used in the inference process. Accuracy is lower when inferring on non-mobile IPs or using them for inference because their homophily is lower (see Table 2). Radius r = 100m.

accuracy by an average of 12% for Oct-2012 and 23% for Feb-2013. The second observation is that the restriction on IAT reduces the number of predictions by an average of 4 times for Oct-2012 and 5 times for Feb-2012.

4.3.5 Inference over IPs with Two or More Known Neighbors

`wvRN(minIAT)` and `wvRN(numMoves)` have different results only when we take into account IPs with two or more known neighbors. To tease out this difference, we ran experiments where IP nodes with one known neighbor were not considered. Table 4.3.5 lists results for IPs with two or more known neighbors. We observe that the accuracy numbers for `wvRN(minIAT)` are higher than `wvRN(numMoves)`. However, these difference are not statistically significant at the 0.05 level.

4.3.6 Accuracy with a Slack Distance

So far, we have considered a correct prediction as one in which the predicted CBG ID is equal to the actual CBG ID. What if we allow some slack in this definition? Suppose we consider a correct prediction as: *the distance between the centroids of the predicted and the actual CBG IDs are within slack kilometers*. Figure 11 shows these results for `wvRN(minIAT)` and different values of slack distance. We observe that in the Oct-2012 data at only 25 km slack (i.e., 15.5 miles), our accuracy is 84%, 86%, and 64%, respectively, for inferring the location of all IPs, non-mobile IPs, and mobile IPs using all neighbors. These accuracy numbers are up from 74%, 77%, and 49%, respectively, when we allowed no slack distance. In the Feb-2013 data at only 25 km slack, our accuracy is 81%, 84%, and 64%, respectively, for inferring the location of all IPs, non-mobile IPs, and mobile IPs using all neighbors. These accuracy numbers are up from 74%, 78%, and 53%, respectively, when we allowed no slack distance. The other combination of runs have similar results and are omitted for brevity.

4.4 Discussion

Figure 12 presents the public IP addresses in our data drawn on a US map. The blue dots are the IPs for which our method inferred the correct CBG ID. The red dots are the IPs for which our method inferred the incorrect CBG ID. Our method tends to perform better in city centers.

Our results showed the following. (1) `wvRN(minIAT)` has slightly better accuracy than `wvRN(numMoves)`, but this difference is not statistically significant at the 0.05 level. (2) Accuracy is not sensitive to the choice of radius

Infer location for **all** IPs with 1 known neighbor with IAT $\leq$ 60min ...	Accuracy wvRN(numMoves) Oct-2012 | Feb-2013	Number of Predictions Oct-2012 | Feb-2013	% Mobile in Predictions Oct-2012 | Feb-2013
using all neighbors	72.7 | 76.0	606,274 | 179,294	92.4% | 87.1%
using mobile neighbors	**75.4** | **79.0**	546,498 | 154,610	94.3% | 90.4%
using non-mobile neighbors	52.6 | 59.2	86,981 | 35,073	74.1% | 68.3%
Infer location for **mobile** IPs with 1 known neighbor with IAT $\leq$ 60min ...			
using all neighbors	74.7 | 78.7	559,976 | 156,226	100% | 100%
using mobile neighbors	**77.1** | **81.9**	515,575 | 139,714	100% | 100%
using non-mobile neighbors	51.3 | 55.9	64,495 | 23,939	100% | 100%
Infer location for **non-mobile** IPs with 1 known neighbor with IAT $\leq$ 60min ...			
using all neighbors	49.1 | 57.5	46,298 | 23,068	0% | 0%
using mobile neighbors	46.9 | 51.5	30,923 | 14,896	0% | 0%
using non-mobile neighbors	**56.3** | **66.5**	22,486 | 11,134	0% | 0%

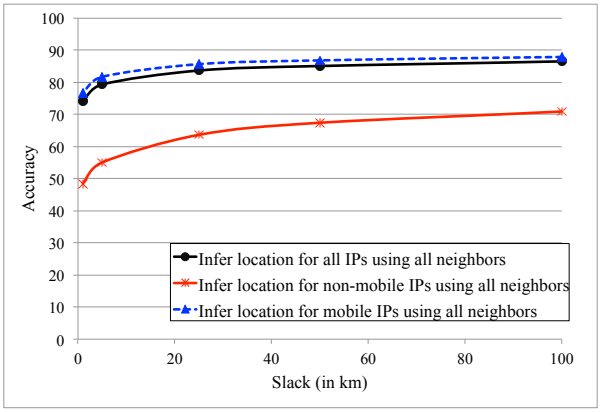
Table 4: Inference over IPs with one known neighbor and IAT $\leq$ 60 minutes. Accuracy results for are shown `wvRN(numMoves)`. The highest accuracy values are in boldface. Restricting IAT to $\leq$ 60 minutes improves accuracy by an average of 12% on Oct-2012 and 23% on Feb-2013 data; but reduces the number of predictions by an average of 4 times for Oct-2012 and 5 times for Feb-2012. See Table 4.3.4 for these differences. Radius r = 100m.

Infer location for **all** IPs with 1 known neighbor with IAT $>$ 60min ...	Accuracy wvRN(numMoves) Oct-2012 | Feb-2013	Number of Predictions Oct-2012 | Feb-2013	% Mobile in Predictions Oct-2012 | Feb-2013
using all neighbors	56.3 | 50.7	1,573,436 | 773,095	81.1% | 78.4%
using mobile neighbors	**59.9** | **55.6**	1,262,022 | 594,504	83.6% | 80.9%
using non-mobile neighbors	42.5 | 35.6	369,519 | 207,099	71.0% | 69.6%
Infer location for **mobile** IPs with 1 known neighbor with IAT $>$ 60min ...			
using all neighbors	59.5 | 55.2	1,275,406 | 605,948	100% | 100%
using mobile neighbors	**63.0** | **61.7**	1,055,269 | 481,151	100% | 100%
using non-mobile neighbors	43.5 | 31.6	262,188 | 144,226	100% | 100%
Infer location for **non-mobile** IPs with 1 known neighbor with IAT $>$ 60min ...			
using all neighbors	42.8 | 34.8	298,030 | 167,147	0% | 0%
using mobile neighbors	**44.2** | 30.0	206,753 | 113,353	0% | 0%
using non-mobile neighbors	40.3 | **44.9**	107,331 | 62,873	0% | 0%

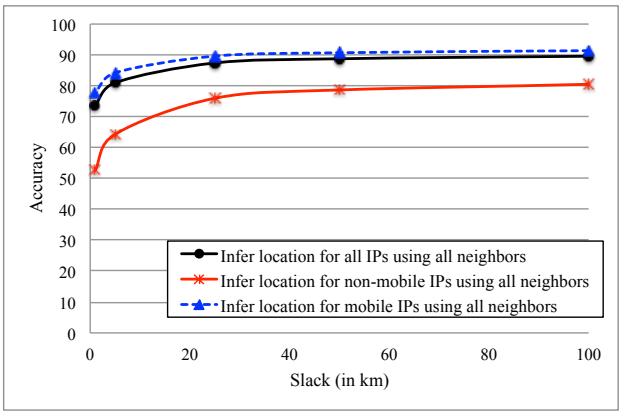
Table 5: Inference over IPs with one known neighbor and IAT $>$ 60 minutes. Accuracy results for are shown `wvRN(numMoves)`. The highest accuracy values are in boldface. Allowing IAT of $>$ 60 minutes decreases accuracy. This is because IAT on the movement edge is correlated with distance. Radius r = 100m.

Infer location for **all** IPs with ≥ 2 known neighbors ...	Accuracy wvRN(minIAT) Oct-2012 | Feb-2013	Accuracy wvRN(numMoves) Oct-2012 | Feb-2013	Number of Predictions Oct-2012 | Feb-2013	% Mobile in Predictions Oct-2012 | Feb-2013
using all neighbors	70.4 | 64.6	66.7 | 61.2	583,405 | 148,721	89.6% | 79.8%
using mobile neighbors	**73.4** | **69.7**	**69.6** | **66.7**	531,146 | 124,064	92.1% | 83.7%
using non-mobile neighbors	46.6 | 48.4	44.8 | 45.5	11,357 | 5,704	26.9% | 24.3%
Infer location for **mobile** IPs with ≥ 2 known neighbors ...				
using all neighbors	73.7 | 70.2	69.7 | 66.9	522,590 | 118,674	100% | 100%
using mobile neighbors	**76.1** | **75.4**	**72.2** | **72.6**	489,026 | 103,916	100% | 100%
using non-mobile neighbors	29.6 | 34.0	26.9 | 30.0	3,058 | 1,384	100% | 100%
Infer location for **non-mobile** IPs with ≥ 2 known neighbors ...				
using all neighbors	42.7 | 42.6	40.1 | 38.6	60,815 | 30,047	0% | 0%
using mobile neighbors	41.9 | 40.4	39.3 | 36.7	42,120 | 20,148	0% | 0%
using non-mobile neighbors	**52.9** | **53.0**	**51.4** | **50.4**	8,299 | 4,320	0% | 0%

Table 6: Inference over IPs with two or more known neighbors: wvRN(minIAT) vs. wvRN(numMoves). For each method, the highest accuracy values are in boldface. The differences between the two methods are not statistically significant at the 0.05 level. The number of predictions varies depending on the particular inference and the neighbor types used in the inference process. Radius r = 100m.

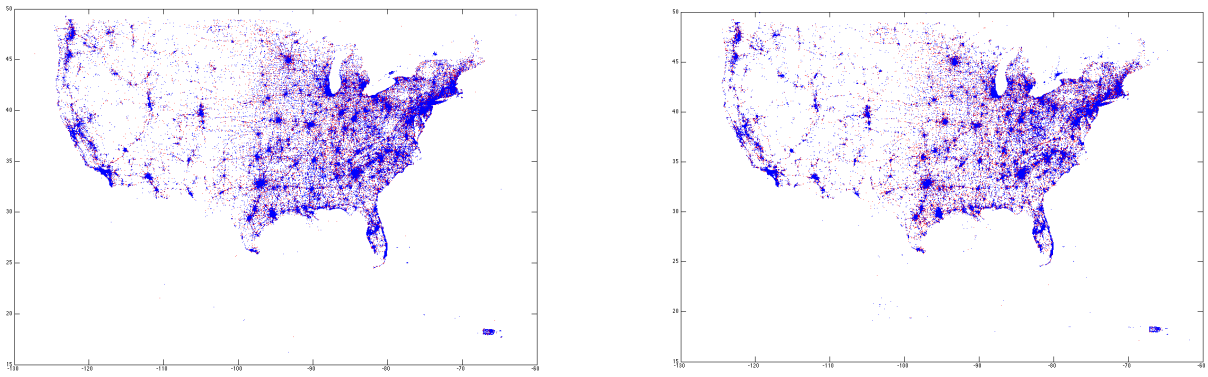


(a) Oct-2012: Accuracy vs. Slack (in km)



(b) Feb-2013: Accuracy vs. Slack (in km)

Figure 11: Allowing slack distance in computing accuracy, where a prediction is considered correct if the distance between the centroids of the predicted and the actual CBG IDs are within *slack* kilometers. At only 25km slack, accuracy increases by an average of 11% in the Oct-2012 data and 9% in the Feb-2013 data. Method used here was `wvRN(minIAT)` and radius r = 100m.



(a) Oct-2012 (b) Feb-2013

Figure 12: Public IP addresses in our data sets drawn on a US map. The blue dots depict the IPs which our method correctly classified (i.e., inferred the correct CBG ID). The red dots depict the IPs which our method incorrectly classified. Our method tends to be more accurate on IPs in urban centers.

r used in determining whether an IP is mobile or non-mobile. Also, based on current router technology, $r = 0.1$ km is a reasonable value. (3) Restricting movement edges to those with smaller IATs increases accuracy but at a sharp reduction in terms of number of predictions. (4) When inferring location on all IPs and on mobile IPs, time-stamped mobile neighbors provide the best information in terms of location. We observe accuracy between 74% and 77% in Oct-2012 and between 75% and 79% in Feb-2013. (5) Inferring location for a non-mobile IP is a hard task because non-mobile IPs tend to have larger inter-arrival times and larger distances than mobile IPs. For example, in the Oct-2012 data 77.1% of the mobile to mobile movements were less than 1 km; while only 44.2% of the non-mobile to non-mobile movements were less than 1 km. Also, number of non-mobile IPs is comparatively much smaller than the number of mobile IPs in the data (see Table 4.3.3. (6) Allowing some slack distance (say of only 25 km) increases accuracy on average by 11% in Oct-2012 and by 9% in Feb-2013.

5 Conclusions

To our knowledge, our work is the first of its kind that uses just the structure of an IP×IP movement graph to infer locations, in terms of CBG IDs, for hashed public IP addresses. We described a novel way to represent hashed IPs in RTB-request data as a heterogeneous movement graph with time-stamped mobile nodes and non-time-stamped non-mobile nodes. The movement edges are annotated with number of movements between two IPs and the inter-arrival distribution. An extensive empirical study on 44M RTB requests showed that using local relational classifiers (such as wvRN) are effective, though the choice of which weight to use on the edge can slightly increase performance. Lastly, we demonstrated that allowing only 25 km (i.e., 15.5 miles) slack distance between the centroids of the predicted and the actual CBG IDs produces an accuracy of over 80% for inferring location on all IP types. With a slack of zero km, the accuracy is 74% for the same inference problem. These results are impressive since we are estimating the correct CBG out of 212K possibilities.

6 Acknowledgments

We thank Jason Dolatshahi and William Payne for helping us collect the data and useful comments. The design of the method was conducted while all authors were at EveryScreenMedia now part of Dstillery.

References

[1] M. Balakrishnan, I. Mohomed, and V. Ramasubramanian. Where's that phone? Geolocating ip addresses on 3G networks. In *Proc. of the 9th ACM SIGCOMM Internet Measurement Conf.*, IMC, pages 294–300, 2009.

[2] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proc. of the 17th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, KDD, pages 1082–1090, 2011.

[3] M. Domenico, A. Lima, and M. Musolesi. Interdependence and predictability of human mobility and social interactions. In *Nokia Mobile Data Challenge: http://research.nokia.com/page/12000*, MDC 2012, 2012.

[4] B. Gallagher, H. Tong, T. Eliassi-Rad, and C. Faloutsos. Using ghost edges for classification in sparsely labeled networks. In *Proc. of the 14th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, KDD, pages 256–264, 2008.

[5] D. Koutra, T.-Y. Ke, U. Kang, D. H. Chau, H.-K. K. Pao, and C. Faloutsos. Unifying guilt-by-association approaches: Theorems and fast algorithms. In *Proc. of the European Conf. on Machine Learning and Knowledge Discovery in Databases*, ECML PKDD, pages 245–260, 2011.

[6] S. A. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research*, 8:935–983, May 2007.

[7] A. Metwally and M. Paduano. Estimating the number of users behind IP addresses for combating abusive traffic. In *Proc. of the 17th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, KDD, pages 249–257, 2011.

[8] S. Scellato, A. Noulas, R. Lambiotte, and C. Mascolo. Socio-spatial properties of online location-based social networks. In *Proc. of the 5th Int'l AAAI Conf. on Weblogs and Social Media*, ICWSM, 2011.

[9] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

[10] Y. Wang, D. Burgener, M. Flores, A. Kuzmanovic, and C. Huang. Towards street-level client-independent ip geolocation. In *Proc. of the 8th USENIX Conf. on Networked Systems Design and Implementation*, NSDI, pages 27–27, 2011.

[11] B. Wong, I. Stoyanov, and E. G. Sirer. Octant: A comprehensive framework for the geolocalization of internet hosts. In *Proc. of the 4th USENIX Conf. on Networked Systems Design and Implementation*, NSDI, pages 23–23, 2007.

