A cheap and accurate delay-based IP Geolocation method using Machine Learning and Looking Glass

Allen Hong*, Yahui Li*, Han Zhang*†, Ming Wang*, Changqing An*, Jilong Wang*†
*Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, China
†Zhongguancun Laboratory, Beijing, China
Emails: allenhong2000@foxmail.com, liyahui_ego@126.com,
m-wang20@mails.tsinghua.edu.cn, zhhan@tsinghua.edu.cn, {acq, wjl}@cernet.edu.cn

***Abstract*—Predicting the geographical location of an IP host is a fundamental and valuable but long-standing challenge in the field of network research. Although delay-based methods have relatively high coverage and low time consumption, currently this type of method is not accurate enough and requires a large number of vantage points, making its cost high. In this paper, we propose a novel delay-based framework to make IP geolocation more accurate and cheap. Firstly, we collect 373 Looking Glass with known geographical addresses and overcome the high cost problem by using them as vantage points. Secondly, we make the prediction of geographical coordinates more accurate by using the machine learning algorithm and regional information of the target IP. Finally, we propose a method based on machine learning to supplement missing values in the delay data and improve the accuracy of geolocation successfully. Our experiment results validate the feasibility and improvement of our method. Using our method, we have an average error of 69.49 km for the geolocation of our test set, which is approximately 160 km less than the state-of-art work.**

***Index Terms*—IP Geolocation; Network Measurement**

I. Introduction

Knowing the geographical location of an Internet host is becoming increasingly important in the digital age as it enables more accurate personal services, such as weather forecasting, targeted advertising, and online copyright management [1]. Although GPS is one such technology, it requires specialized hardware on the client side, making it inaccessible for many users. In order to address this challenge, IP geolocation, which is based on a host's IP address, provides a client-independent method for determining geographical location without requiring any support from the client.

There have been many IP geolocation methods in recent years, which can roughly be divided into passive and active. Passive methods rely on information about an IP address, such as its hostname, to infer its geographical location [2]–[4]. On the other hand, active methods use machines to actively send packets to the target IP and predict its location based on the measurement data [5]–[7]. There is no clear superiority between these two methods, as they are respectively suitable for different application scenarios. In this article, we focus on improving active methods. Compared with passive methods, active methods have *broader coverage* since they can geolocate an IP as long as it is able to reply to the packets. They also have *strong timeliness* as active probing when geolocating avoids outdated source data effectively. Some active methods use traceroute to collect topology to geolocate. However, the use of traceroute can result in increased time and network traffic, causing these methods *low scalability*. Another active method, called delay-based, uses only the ping command and attempts to geolocate based on latency. This method is less accurate but requires less time and is suitable for scenarios where time and coverage are a high priority, but accuracy is less critical. However, there are still some challenges associated with delay-based methods.

The first challenge with delay-based methods is their high cost, as they require multiple controlled machines to ping the target. Traditional approaches involve using cloud servers or measurement platform probes [8], which can be relatively expensive. In this paper, we take advantage of the Looking Glass (LG) to address this challenge. LG are tools deployed on routers by some network operators and some of them can be accessed through webpages, making them a more cost-effective solution. We collect 373 LG with known locations that can be automatically used based on a public list from [9]. These LG can be used for geolocation purposes and can also be applied to other research that requires location-known machines.

Another challenge with delay-based methods is the difficulty of converting latency to geographical location, as latency can be influenced by many factors such as geographical distance, physical media, and congestion level, meaning the collected delay data often contains significant amounts of noise. We pinpoint that it is suitable to use machine learning algorithms to address this challenge. After comparing various common machine learning algorithms, we choose the Random Forest as the algorithm for geolocating. We also discover that including the target IP's regional information when geolocating can further improve accuracy. The last challenge is the presence of missing values in the measurement data, which can render certain algorithms unavailable. To address this, we propose a novel method based on the XGBoost model to supplement missing data. To the best of our knowledge, this is the first method to address this issue in the IP geolocation.

This work is partially supported by the National Key Research and Development Program (No. 2020YFE0200500) and the National Science Foundation (No. 62102020, No. 62002009).

Yahui Li and Han Zhang are the co-corresponding authors.

ISBN 978-3-903176-57-7

To evaluate our method, we conducted a large-scale experiment. We used 175 Looking Glass to ping 17,276 terminal addresses and 26,625 router addresses with known locations. The results of the experiment demonstrate the feasibility of using Looking Glass for IP geolocation, achieving an average error of 69.49 km for the IPs we collected. This represents a significant improvement over previous methods, as the best result achieved by previous techniques had an average error of 230.40 km. Furthermore, the results showed that our novel method for dealing with missing values in measurement data reduced the mean error by 20 km. In summary, the main contributions of this work are as follows:

- We have collected 373 Looking Glass with known locations, which can be leveraged for IP geolocation purposes and for other research that requires access to machines with known positions for packet transmission.
- We have proposed a systematic approach based on Random Forest to improve the accuracy of delay-based IP geolocation methods. This approach resulted in a mean error reduction of 160 km when compared to the best method in previous studies.
- We have addressed the issue of missing measurement data, which is a common challenge in delay-based geolocation methods, by introducing a novel method based on XGBoost, which has not been previously studied.

II. Related Works

Previous works of IP geolocation can be classified into two categories: passive geolocation and active geolocation. The difference between them mainly lies in whether to send packets to the target IP actively.

A. Passive Geolocation

Passive geolocation tries to find some information about an IP which can be used to infer geographical location. Some use RDNS hostnames to geolocate because network operators may encode city names or other location hints in hostnames [2]–[4]. Wang et al. found that some entities host web services locally. They mined this information through the mapping service [10]. Guo et al. presented an approach called Structon, which extracts location from the web content of some IPs [11]. Liu et al. proposed Checkin-Geo which leverages the location data that users share in social networks [12]. Wang et al. discovered that the source code of some websites hosting live webcams exposes the IP address and corresponding location [13]. They used the natural language processing technique to extract these web pages and got the location of some IPs. Passive methods are fast, but they fail to apply to too many addresses because few IP addresses have information that can be used to infer. In this paper, we choose some of these methods to generate some IP addresses with known location which we will show in section III-C.

B. Active Geolocation

Active geolocation uses some machines with known locations to send packets to the target. The sender is called the vantage point, and the receiver is called the end point. Depending on whether only using ping to send packets, we can split it into two categories: delay-based and topology-used.

1) delay-based: These methods only use ping to send packets and use delay obtained to geolocate. This type of method has a great coverage because it can geolocate any IP as long as this IP can respond to ping. Some methods geolocate by a process called *multilateration*, which converts latency to a physical distance for each vantage point and then draws circles around each vantage point with its corresponding distance as a radius [5]–[7]. The target must be in the area where these circles intersect, and its geographical coordinate is marked as the centroid of that area. The main problem with multilateration is that it is hopeless to predict accurate physical distance by latency. Many factors can influence latency, such as circuitous routing path, processing delay, etc. Much of the research on active focuses on increasingly complex models. Eriksson et al. converted the prediction to a classification problem [14]. They divided the region into several categories and use naive Bayes to predict the category of the target. But the accuracy is not good when predicting in a large area because the number of categories is too much. Jiang et al. trained a two-stage neural network to estimate the geolocation [15]. Their method can not be used in practice because its neural network does not adapt for geolocation, leading prediction not accurate. In summary, the accuracy of these delay-based methods still needs to be improved.

2) topology-used: Some methods not only use ping but also traceroute for sending packets to end points. In other words, these methods combine network delay and network topology to infer location [1], [16], [17]. These methods are more accurate than delay-based methods. However, there are three drawbacks to them. First, using traceroute additionally takes more time than just using ping. Second, traceroute causes too much network traffic, making it impossible to extend the method to the entire Internet. Finally, some IP addresses deny only traceroute traffic causing these addresses cannot to be inferred. To summarize, there are problems with efficiency and coverage in this type of method.

III. Datasets

In this section, we describe our process of building datasets. First, we constructed a location hint dictionary, which is used to build some other datasets in our study. Next, we collected Looking Glass with known locations to serve as our vantage points. We obtained endpoint data from some public resources such as RDNS hostname dataset. Finally, we use vantage points to ping the end points, recording the minimal round-trip time (RTT) between each pair. This RTT data was compiled into our delay dataset.

A. Location Hints Dictionary

The location hints dictionary returns the geographical coordinate of the corresponding location when entering a location hint. We obtain city information from the *cities 1000* file in the June 2022 snapshot of a free geodatabase called GeoNames

[18]. This file contains information about all cities with a population over 1000, such as name, population, and geographic coordinates. The city name is used as the location hint, i.e., the key of the dictionary. In addition to it, we use three other types as location hints, which network operators may sometimes use to represent cities:

CLLI: This 11-character code is used in the North American telecommunications industry, and the first six characters specify the location. We get a CLLI code list from [19].

UN/LOCODE: It is a code representing trade and transport locations. We use the dataset released by United Nations Economic Commission for Europe (UNECE) [20].

IATA and ICAO: Both of them are the airport code, which is used to designate airports. Some operators prefer to use them to represent the city where the airport is located. We collected these codes from the OurAirports database [21].

Although some location hints can be used to represent a city, network operators may use them for other purposes. For example, the word *normal* can represent the city *Normal, Illinois, USA*, but the operator probably uses it to indicate that the device is normal. To solve it, we manually build a blacklist of such words that may cause false representation.

B. Vantage Points

Previous methods of active IP geolocation have relied on expensive vantage points, such as cloud servers or probes from measurement platforms like RIPE Atlas. However, our approach offers a more cost-effective solution. By using Looking Glass as our vantage points, we are able to significantly reduce the cost of geolocation. LG are tools that are deployed on routers by network administrators for diagnostics, and many can be accessed through a web interface at a minimal cost. In addition to being cost-effective, LG ,which is deployed on the router, can offer a more stable network environment than many of the terminal probes used in previous methods. In Appendix A, we check that whether the accuracy is reduced when using LG as vantage points.

In this paper, we collected a set of 373 Looking Glass with known locations that can be used automatically for IP geolocation. To obtain the LG, we crawled the source code of the webpages of each LG in a list published by [9]. We analyzed the structure of the source code and extracted possible geographical location keywords using regular expressions. For example, we looked for phrases in the form of 'X, Y' where Y is the name of a country or state (we found that only the US, Canada, and Brazil used state name as Y). We considered X as a potential geographical name and included it in our list of candidates. We then matched these candidates against our location hint dictionary to obtain the corresponding geographical coordinates for each LG. Next, we analyzed the API of each LG and checked for availability to ensure that it could be used for geolocation. Although there are already public available LG [9], [22], their geographic locations are derived from IP geolocation databases, which can compromise their accuracy. The locations of our LG were obtained by analyzing the content of their corresponding webpages, providing a more accurate and reliable source of information. Fig. 1 shows the geographical locations of the LG we collected.

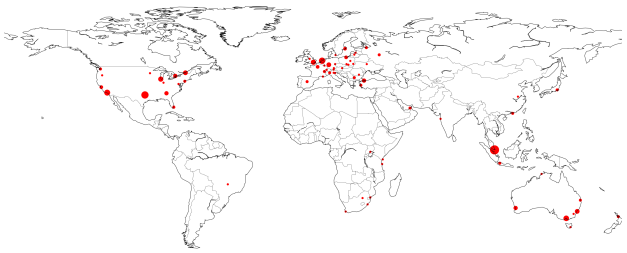


Fig. 1: Geographical location of vantage points

C. End Points

We obtain our end points with two types: terminal address and router address.

1) terminal address: To collect terminal addresses, we used two sources. The first source was the probes from the RIPE Atlas platform, from which we obtained 23,746 addresses. For the second source, we used the GeoCAM method [13], which found that the source code of some camera-hosting websites could expose the IP address. To this end, we parsed two websites, *pictimo.com* and *insecam.org*, collected 21,343 IP addresses along with their corresponding city names and used the location hint dictionary to obtain the corresponding geographical coordinates finally.

2) router address: To obtain router addresses, we used the RDNS-based method described in Section II-A. Specifically, we used the November 2021 version of the RDNS dataset maintained by Rapid7 [23], which contains reverse DNS hostnames for all non-blacklisted and non-private IPv4 addresses. Some regular expressions for the hostnames were publicly released by [4], which we used to obtain the location hint for the corresponding hostname. In addition, we defined our own regular expressions. If the rightmost label of a hostname is an ISO-3166 country code and includes a full city name in that country, we geolocated the hostname to that city. We considered a /24 subnet to be *geo-good* if we geolocated over 200 IP addresses to the same city. To construct our router address set, we used the 2021 version of the CAIDA ITDK dataset [24]. We added a router IP to our set if its /24 prefix was geo-good. Our rules and those from [4] yielded a total of 1,058,016 router addresses.

In this paper, 175 vantage points are used to ping a selection of end points. The reason for not utilizing all collected LG is that in some areas, the number of LG is relatively high, which may lead to bias in the measurement data. The minimal RTT between each pair is recorded. End points that violate the rule that data travels through fiber optic cables at almost 2/3 the speed of light in a vacuum are removed [16]. This results in an end point dataset with 26,625 router addresses and 17,276 terminal addresses. Table I shows the distribution of these end points in each Regional Internet Registry (RIR).

TABLE I: Number of end points in 5 RIRs

	APNIC	RIPE NCC	LACNIC	ARIN	AFRINIC
Terminal IP	4126	10108	443	2568	31
Router IP	4606	14157	2637	5225	0

IV. Analysis

A. Is IP geolocation database accurate?

Commercial IP geolocation databases are frequently used by researchers to determine the location of an IP address. However, previous studies have shown that these databases may not be accurate enough [25]. Since our end points are obtained by some geolocation methods, which are also the principle of databases, using them will lead to highly biased results. To re-evaluate these databases based on our data, we used LG as the evaluation standard, since we geolocated them by analyzing their webpage content, which differs from how the databases obtain their information. We evaluated the free versions of two commonly used databases, IP2Location and MaxMind [26], [27]. The results showed mean errors of 1376.00 km and 1098.34 km, respectively. Additionally, their country-level accuracy was found to be 80.16% and 82.31%. These findings indicate that the accuracy of these databases, even at the country level, is questionable. These results prove the necessity of the IP geolocation method.

B. Is missing RTT inevitable?

As we all know, some packets might be lost in transmission , so we cannot ensure successfully getting the RTT of each pair. During our data collection, we also encountered some pairs do not have RTT even though we make ping several time. In our first data collection, 8.61% of pairs did not have RTT. We attempted to obtain RTT for these pairs by having the vantage points in these pairs ping the corresponding endpoints five times, but 29.8% of the pairs still did not have RTT. We explain that it may be due to *geoblocking*, a technology restricting access to Internet content based on the sender's geographical location. This missing data can reduce the accuracy of geolocation methods that rely on RTT and can also slow down the geolocation process due to the need to re-ping. We propose a method to supplement the missing RTT data, which improves the accuracy of geolocation.

C. Is traveling speed related to region?

For convenience, we call the *traveling speed* by dividing the geographical distance by half of the RTT. Previous delay-based methods did not consider any factors affecting the traveling speed. However, many factors affect the traveling speed, such as physical media, congestion level, and geographic region. Among these factors, the geographical region can be inferred easily. Before geolocating, we analyze the relationship between the geographical region and traveling speed. Fig. 2 shows the traveling speed of vantage points in the United States (US), Singapore (SG), Australia (AU), and Russia (RU) to some different regions. We can see that the traveling speed of vantage points to end points in different regions varies significantly, and that the speed of a vantage point to end points in a specific region varies across different regions. For example, the speed of most US, SG, and AU vantage points to western Europe end points is between 100 and 150 km/ms while most RU vantage points to these end points have traveling speeds between 50 and 100 km/ms. Based on this analysis, we incorporated the regional information of the target IP address when geolocating by RTT.

D. Is using Looking Glass feasible?

Now we explore the advantages of using LG as vantage points compared with the widely-used RIPE Atlas platform for geolocation measurements. We employed 175 LG to ping about 40,000 endpoints, which would require approximately 42,000,000 credits in RIPE Atlas. To earn credits, one can either become a sponsor or host a probe to earn credits. However, sponsoring 50,000 EUR would be necessary to obtain such a number of credits, while hosting a probe can earn about 25,000 credits per day, meaning that completing this measurement would take 2000 days if credits were earned by hosting one probe. Our estimation shows that LG reduce the cost of such measurements significantly. In addition to reducing costs, LG can also expand geographical coverage, as the 175 LG we used are distributed across 75 cities, including 25 that are not covered by RIPE Atlas. Our aim is to offer more choices for researchers looking for vantage points with known positions, rather than replacing existing measurement platforms. We believe that our published Looking Glass can be a valuable resource for researchers seeking more affordable and diverse measurement options.

V. Method

A. Overview and defined symbols

Fig. 3 illustrates the framework of our method, which is comprised of three stages. In stage 1, we predict the missing data in the measurement data whose details are described in Algorithm 1. Random Forest is used in stage 2 to predict regions, followd by the predition of geographical coordinates in stage 3. We now define some symbols later used. Several symbols are used later, including K to represent the number of vantage points, N for the number of end points in the training set, and M for the number of end points in the test set. The RTT matrix, denoted as $\mathbf{X}$, is defined as the matrix with each row representing the RTT between one end point and each vantage point. The size of $\mathbf{X}_{train}$, which represents the RTT matrix of the training set, is $N \times K$, while the size of $\mathbf{X}_{test}$, the RTT matrix of the test set, is $M \times K$.

B. Stage 1: infer missing RTT

The presence of missing values can adversely affect machine learning models, especially those that cannot handle missing data, such as fully connected neural networks. Additionally, traditional methods of replacing missing values with median or mean may not be optimal for the IP geolocation problem. We claim that the succefully obtained delay between vantage points and end points can help to infer missing RTT

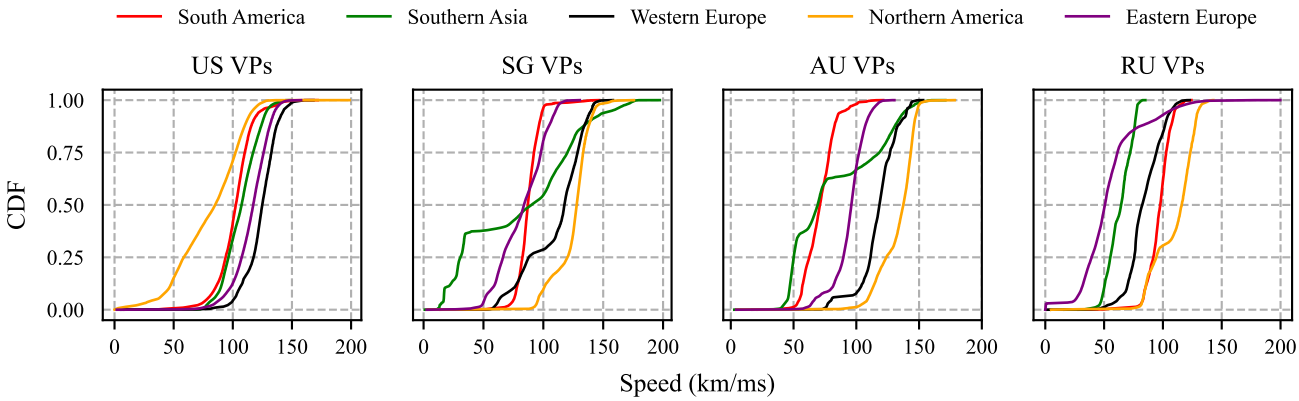


Fig. 2: Speed between vantage points in some countries and end points in some regions

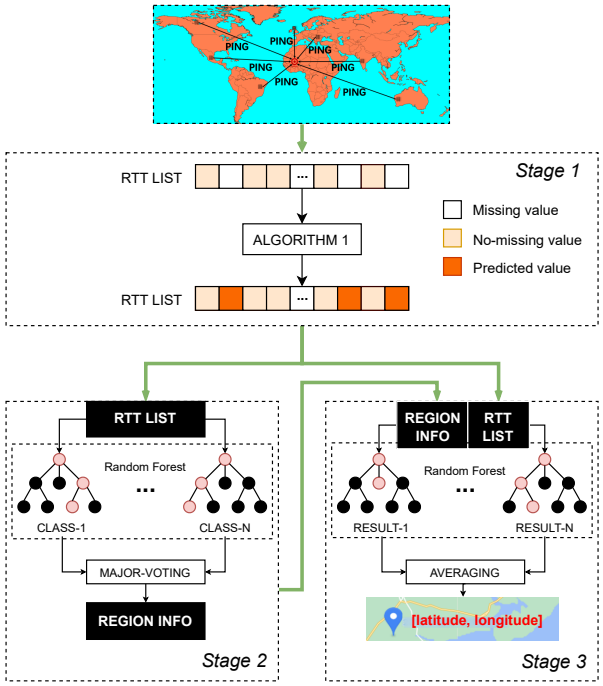


Fig. 3: Framework of our method

Algorithm 1 Handle missing values

Input: RTT Matrices $\mathbf{X}_{train}$ and $\mathbf{X}_{test}$
Output: New RTT Matrices $\mathbf{NX}_{train}$ and $\mathbf{NX}_{test}$

$M \leftarrow$ Array(size: K)
for $col \leftarrow 1$ to K **do**
  Initialize two empty arrays TX and TY
  for $row \leftarrow 1$ to N **do**
    if $\mathbf{X}_{train}[row, col]$ is not none **then**
      $TX.append(\mathbf{X}_{train}[row].RemoveIndex(col))$
      $TY.append(\mathbf{X}_{train}[row, col])$
  $M[col] \leftarrow$ XGBoost trained with TX and TY

$\mathbf{NX}_{train} = \mathbf{X}_{train}$, $\mathbf{NX}_{test} = \mathbf{X}_{test}$
for $col \leftarrow 1$ to K **do**
  for $row \leftarrow 1$ to N **do**
    if $X_{train}[row, col]$ is none **then**
      $L \leftarrow X_{train}[row].RemoveIndex(col)$
      $\mathbf{NX}_{train}[row, col] = M[col].predict(L)$
  for $row \leftarrow 1$ to M **do**
    if $\mathbf{X}_{test}[row, col]$ is none **then**
      $L \leftarrow \mathbf{X}_{train}[row].RemoveIndex(col)$
      $\mathbf{NX}_{test}[row, col] = M[col].predict(L)$
return $\mathbf{NX}_{train}$, $\mathbf{NX}_{test}$

values. The details of this stage are described in Algorithm 1. It comprises K rounds, where in each round, we supplement the RTT between one specific vantage point and end points. That is to say, we predict the missing value of one column of the RTT matrix in each round. For round i, we select the rows whose ith element is not null in $\mathbf{X}_{train}$, and use the data from these rows to train a machine learning model. The model takes the data of all columns except column i as input, and the data of column i as output. Once the model is trained, it is used to predict missing values for both $\mathbf{X}_{train}$ and $\mathbf{X}_{test}$. We use XGBoost as the machine learning algorithm in this stage. XGBoost is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library [28]. The reason of choosing this algorithm is that it supports missing values by default so the missing values of some columns of data can be dealt with well in the training stage. After stage 1, $\mathbf{X}_{train}$ and $\mathbf{X}_{test}$ no longer have any missing elements.

C. Stage 2: infer region of target

In stage 2 of our method, our objective is to predict the region of an IP address. While RIRs may provide regional information, their data can be outdated, and IP geolocation databases may contain incorrect passive source information. To address these limitations, we propose using RTT measurements to predict the region, which is a classic classification problem. Specifically, our method formulates the problem as $r = \mathbf{F}(\mathbf{L})$, where r is the predicted region, $\mathbf{F}$ is the machine learning algorithm, and L represents the RTTs between the target and our vantage points. Due to the complexity of the relationship between delay and geographical distance, as well

as the presence of noisy data, we adopt the Random Forest algorithm in our approach, which is known for its robustness to outliers and non-linear data. Further details on our algorithm selection and a comparison of accuracy with other machine learning algorithms are provided in Appendix B.

D. Stage 3: infer geographical coordinate

In stage 3, we incorporate regional information along with RTT to improve the accuracy of IP geolocation. Specifically, we add an extra dimension to the input by including the predicted region from stage 2. The output is the geographic coordinates of the IP, namely latitude and longitude. To be more precise, we modify the loss function by using the great-circle distance, which is the shortest distance between two points on the surface of a sphere. We denote the predicted geographical coordinate as y, with y^0 representing the latitude and y^1 representing the longitude. For two geographical coordinates y_i and y_j, their distance Δd can be calculated:

$$\Delta d = 2l * \arcsin\sqrt{\sin^2 \Delta lat + \cos y_i^0 * \cos y_j^0 * \sin^2 \Delta lon}$$

where

$$\Delta lat = y_i^0 - y_j^0$$
$$\Delta lon = y_i^1 - y_j^1$$

The symbol l represents the arithmetic mean radius of earth which is 6,371.0088 km [29]. Because the calculation of great-circle distance requires a lot of trigonometric function operations which will take too much time, we adjust the format of (latitude, longitude) to n-vector which can be converted by:

$$\mathbf{n}_i = \begin{bmatrix} \cos(y_i^0) * \cos(y_i^1) \\ \cos(y_i^0) * \sin(y_i^1) \\ \sin(y_i^0) \end{bmatrix}$$

where $\mathbf{n}_i$ represents the n-vector of y_i. After this conversion, the geographical distance can be calculated as:

$$\Delta d = 2l * \arctan\left(\frac{|\mathbf{n}_i \times \mathbf{n}_j|}{\mathbf{n}_i \cdot \mathbf{n}_j}\right) \quad (1)$$

For one IP, the problem can be formulated as:

$$\mathbf{n} = \mathbf{ML}(L, r)$$

where **n** is the n-vector format of the target, **ML** is the machine learning algorithm of this stage, L is the RTTs between this target and our vantage points, and r is the category of the target's region which is inferred in stage 2.

We choose to use the Random Forest for geolocation as same as stage 2 because of its good robust to outliers. In addition, the problem of this stage is a multi-output regression problem where the size of output is not one-dimensional. Previous work shows that the Random Forest algorithm is suitable for this problem [30]. From the equation (1), the loss function of this Random Forest can be set as:

$$\begin{aligned} L &= \frac{1}{N}\sum_j \Delta d(\mathbf{n}_j, \overline{\mathbf{n}_j}) \\ &= \frac{2r}{N}\sum_j \arctan\left(\frac{|\mathbf{n}_j \times \overline{\mathbf{n}_j}|}{\mathbf{n}_j \cdot \overline{\mathbf{n}_j}}\right) \end{aligned}$$

In practical use, we drop the fixed coefficient $2r$ to improve the speed. In section VI-C, we compare the impact of our loss function adjustment on the accuracy of geolocation. Our results demonstrate that the adaption of the loss function significantly improves the accuracy. Furthermore, we also successfully improved the speed of conversion from geographic coordinates to n-vector format.

VI. Evaluation

In this section, we evaluate our method in four aspects. First, we evaluate the impact of our missing value handling method on the accuracy. Next, we explore the importance of region information in our method. Then, we evaluate the impact of our loss function used in stage 3. Finally, we compare the accuracy of our method with prior works. For evaluation, we randomly divide end points into 80% for training and 20% for testing. It should be noted that the accuracy heavily depends on the test data, i.e., the IP address being geolocated. The accuracy of the same method is quite inconsistent in different papers. Our experiment may get different results compared with other papers. We have made our code and data publicly available for further research [31].

A. Handling missing values

In stage 1 of our framework, we propose a novel approach to handling missing values in RTT data. We compare our approach to the traditional methods of keeping missing, imputing with the mean, and imputing with the median. Keeping missing means using models that support missing values instead of supplementing the missing data. We use the XGBoost algorithm, which supports missing values by default, to train and predict using RTTs as input and geographical coordinates as output. Fig. 4 depicts the CDF curve of the error distance when using all end points, which shows that our method outperforms the other methods in terms of error distance. The average error of our method is about 20km smaller than that of other methods. We also evaluate our method under different degrees of data deletion. We select 150 vantage points and 41,046 end points, which guarantees no missing RTT between them. Then we remove some recorded RTT randomly and use different ways of handling missing to supplement data. We use stage 2 and stage 3 of our method to geolocate finally. We find that our filling method successfully improves geolocation accuracy, as shown in Fig. 5. These results validate the necessity of stage 1 in our framework and to our knowledge, this is the first proposed solution to address the problem of missing values of RTT in active IP geolocation.

B. Utilizing region information

In this subsection, we investigate the impact of incorporating regional information in stage 2 of our three-stage geolocation method. We evaluate four different classification types: (1) None, which means not using region information; (2) RIR; (3) Continent; (4) Subregion, which is from Untied Nations Statistics Division's geoscheme [32]. Table II shows the prediction errors of using different ways of dividing the world

TABLE II: Average Error (km) when using different region information

Number \ Type	None	RIR	Continent	Subregion
2500	237.96	**203.54**	231.81	225.58
10000	139.27	132.53	135.92	**124.61**
17500	93.01	94.23	93.02	**88.96**
25000	93.59	88.32	87.58	**81.32**
32500	82.79	81.13	80.52	**75.07**
40000	78.23	76.25	75.03	**70.32**

and different numbers of end points. The results demonstrate that the use of regional information consistently improves the accuracy of our method, regardless of the classification type. Using the Subregion classification method yields the best results, except when the number of end points for training is limited. Therefore, we select the Subregion classification method for stage 2 of our method. We further validate the plausibility of the stage 2 predictions by observing a high accuracy rate of 99.27% when using all end points with the Subregion classification. These results support the necessity of stage 2 and indicate that the inferred results from stage 2 can be reasonably used in stage 3.

C. Customizing the loss function

In stage 3 of our framework, we developed a customized loss function that was specifically tailored for the IP geolocation problem. We compared the performance of two common loss functions, mean absolute error (MAE) and mean square error (MSE), with our custom loss function in Table III. Our results show that using the loss function adapted by great circle distance achieves the highest accuracy in all our tests. In the best case, where 17,500 end points were used, the custom loss function reduced the error by approximately 5km. These results demonstrate the improvement caused by our defined loss function in the IP geolocation problem. Additionally, we compared the time consumption of using geographic coordinates and using (1) to calculate the loss

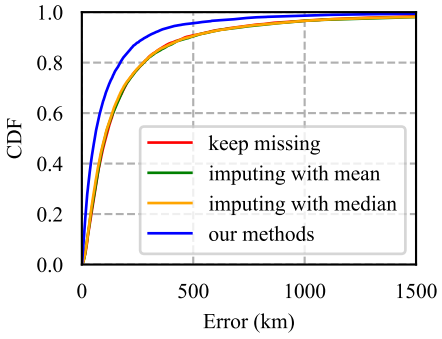


Fig. 4: The CDF curve of geolocation error using different filling methods

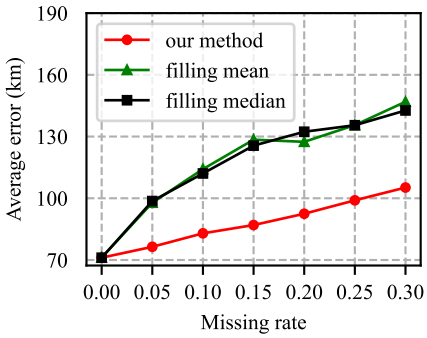


Fig. 5: Comparison of methods for filling missing under different degrees of deletion

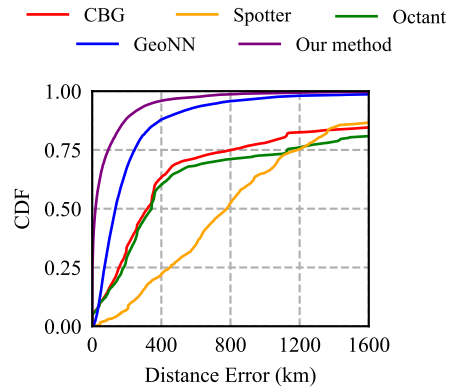
TABLE III: Average Error (km) when using different loss function

Number \ Loss function	MAE	MSE	Our defined
2500	250.60	236.82	**228.47**
10000	137.24	139.16	**135.92**
17500	98.06	93.37	**89.84**
25000	93.59	88.32	**87.58**
32500	99.29	92.83	**89.46**
40000	90.13	83.07	**81.23**

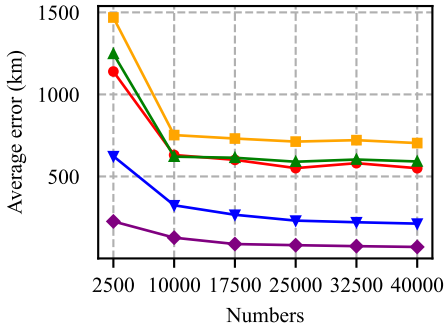
function. Our findings show that converting the geographical coordinate to the form of n-vector can reduce the time by a factor of ten, indicating the effectiveness of this conversion for the IP geolocation problem.

D. Accuracy comparison with prior works

To demonstrate the improvement of our three-stage framework, we compare our method with some prior works. We choose the following baselines to compare: CBG [5], Octant [6], Spotter [7], and GeoNN [15]. The first three methods convert RTT to physical distance and then predict by *multilateration*, which we explain in section II-B. The difference between them is the function of converting RTT to physical distance. The last method, GeoNN, trains a two-tier neural network to make a prediction. Fig. 6a shows the experiment results using our total end points. We achieved the best performance, with a mean error of 69.49 km in geolocating IPs in our test set, compared to the best previous work, GeoNN, which had an error level of 230.40 km. Our method also achieved the best geolocation results in every test, as shown in Fig. 6b, validating its success in improving delay-based IP geolocation accuracy. 36.19% of the addresses had a prediction error of fewer than 10 km, and 60.36% had a prediction error of fewer than 50 km. These results confirm the superiority of our proposed method.



(a) The CDF curve of prediction error when using all end points



(b) Average error when using different numbers of used end points

Fig. 6: Comparison of different delay-based IP geolocation methods

VII. Conclusion and Future work

In this paper, we presented a novel framework for IP geolocation that utilizes a new dataset of 373 Looking Glass as vantage points and incorporates machine learning and regional information to improve accuracy while reducing cost. We addressed the issue of missing RTT data in geolocation and demonstrated the effectiveness of our approach through experiments, achieving superior results compared to prior works. Our code and data are publicly available for further research [31], and we hope our work will contribute to the development of more efficient and cost-effective IP geolocation methods.

In the future, we will use all collected LG to check whether the geographical imbalance of the vantage points have an impact on the accuracy. Additionally, the geographic distribution of the end points we used is also unbalanced, resulting in better performance in areas with a higher number of end points. While this is reflective of the distribution of real-life IP addresses, it is also necessary to focus on improving the geolocation accuracy in areas with a lower number of IP addresses. We plan to further analyze these areas in order to improve overall accuracy.

Acknowledgment

This research is supported by the National Key Research and Development Program of China, 2020YFE0200500.

References

[1] C. Xiang, X. Wang, Q. Chen, M. Xue, Z. Gao, H. Zhu, C. Chen, and Q. Fan, "No-jump-into-latency in china's internet! toward last-mile hop count based ip geo-localization," in *Proceedings of the International Symposium on Quality of Service*, 2019, pp. 1–10.

[2] B. Huffaker, M. Fomenkov, and K. Claffy, "Drop: Dns-based router positioning," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 5–13, 2014.

[3] Q. Scheitle, O. Gasser, P. Sattler, and G. Carle, "Hloc: Hints-based geolocation leveraging multiple measurement frameworks," in *2017 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 2017, pp. 1–9.

[4] M. Luckie, B. Huffaker, A. Marder, Z. Bischof, M. Fletcher, and K. Claffy, "Learning to extract geographic information from internet router hostnames," in *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*, 2021, pp. 440–453.

[5] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida, "Constraint-based geolocation of internet hosts," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, 2004, pp. 288–293.

[6] B. Wong, I. Stoyanov, and E. G. Sirer, "Octant: A comprehensive framework for the geolocalization of internet hosts." in *NSDI*, vol. 7, 2007, pp. 23–23.

[7] S. Laki, P. Mátray, P. Hága, T. Sebők, I. Csabai, and G. Vattay, "Spotter: A model based active geolocation service," in *2011 Proceedings IEEE INFOCOM*. IEEE, 2011, pp. 3173–3181.

[8] "Ripe atlas measurement platform." [Online]. Available: https://atlas.ripe.net/

[9] S. Zhuang, J. H. Wang, J. Wang, Z. Pan, T. Wu, F. Li, and Z. Zhang, "Discovering obscure looking glass sites on the web to facilitate internet measurement research," in *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies*, 2021, pp. 426–439.

[10] Y. Wang, D. Burgener, M. Flores, A. Kuzmanovic, and C. Huang, "Towards {Street-Level}{Client-Independent}{IP} geolocation," in *8th USENIX Symposium on Networked Systems Design and Implementation (NSDI 11)*, 2011.

[11] C. Guo, Y. Liu, W. Shen, H. J. Wang, Q. Yu, and Y. Zhang, "Mining the web and the internet for accurate ip address geolocations," in *IEEE INFOCOM 2009*. IEEE, 2009, pp. 2841–2845.

[12] H. Liu, Y. Zhang, Y. Zhou, D. Zhang, X. Fu, and K. Ramakrishnan, "Mining checkins from location-sharing services for client-independent ip geolocation," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 619–627.

[13] Z. Wang, Q. Li, J. Song, H. Wang, and L. Sun, "Towards ip-based geolocation via fine-grained and stable webcam landmarks," in *Proceedings of The Web Conference 2020*, 2020, pp. 1422–1432.

[14] B. Eriksson, P. Barford, J. Sommers, and R. Nowak, "A learning-based approach for ip geolocation," in *International Conference on Passive and Active Network Measurement*. Springer, 2010, pp. 171–180.

[15] H. Jiang, Y. Liu, and J. N. Matthews, "Ip geolocation estimation using neural networks with stable landmarks," in *2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2016, pp. 170–175.

[16] E. Katz-Bassett, J. P. John, A. Krishnamurthy, D. Wetherall, T. Anderson, and Y. Chawathe, "Towards ip geolocation using delay and topology measurements," in *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, 2006, pp. 71–84.

[17] Z. Wang, F. Zhou, W. Zeng, G. Trajcevski, C. Xiao, Y. Wang, and K. Chen, "Connecting the hosts: Street-level ip geolocation with graph neural networks," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 4121–4131.

[18] "The geonames geographical database." [Online]. Available: https://download.geonames.org/export/dump/

[19] "The clli code list." [Online]. Available: http://wedophones.com/Manuals/TelcoData/clli-lat-lon.txt

[20] "The un/locode list." [Online]. Available: https://unece.org/trade/cefact/unlocode-code-list-country-and-territory

TABLE IV: Average error (km) of using the Looking Glass and the RIPE Atlas probes

	1	2	3	4	5
RIPE Atlas	86.84	115.83	122.57	88.47	94.82
Looking Glass	87.86	90.83	102.77	93.15	91.94

[21] "The iata/icao list." [Online]. Available: https://ourairports.com/data/

[22] V. Giotsas, A. Dhamdhere, and K. C. Claffy, "Periscope: Unifying looking glass querying," in *International Conference on Passive and Active Network Measurement*. Springer, 2016, pp. 177–189.

[23] "Rapid7 rdns dataset." [Online]. Available: https://opendata.rapid7.com/sonar.rdns_v2/

[24] "Caida's traceroute dataset." [Online]. Available: https://publicdata.caida.org/datasets/topology/ark/ipv4/probe-data/team-1/

[25] M. Gharaibeh, A. Shah, B. Huffaker, H. Zhang, R. Ensafi, and C. Papadopoulos, "A look at router geolocation in public and commercial databases," in *Proceedings of the 2017 Internet Measurement Conference*, 2017, pp. 463–469.

[26] "Ip2location lite database." [Online]. Available: https://lite.ip2location.com/ip2location-lite

[27] "Maxmind geolite2 database." [Online]. Available: https://dev.maxmind.com/geoip/geolite2-free-geolocation-data?lang=en

[28] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[29] H. Moritz, "Geodetic reference system 1980," *Journal of geodesy*, vol. 74, no. 1, pp. 128–133, 2000.

[30] H. Borchani, G. Varando, C. Bielza, and P. Larranaga, "A survey on multi-output regression," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 5, no. 5, pp. 216–233, 2015.

[31] "The code and data of this paper." [Online]. Available: https://github.com/activeGeo/activeGeo

[32] "United nations geoscheme." [Online]. Available: https://unstats.un.org/unsd/methodology/m49/

APPENDIX

A. Looking Glass vs Terminal Probes

In this article, we utilize the Looking Glass to reduce the cost in IP geolocation. Since many LG are deployed on routers, their network conditions are more stable than the probes installed in the terminal. We check whether using the LG can get better accuracy than terminal probes. We use the RIPE Atlas platform to get the terminal probes. We filter out 50 geographical locations, of which both the RIPE Atlas platform and our Looking Glass list have probes on that location. Then we select a probe as our vantage points from the LG and the RIPE Atlas at each of these geographical addresses. We select 5,000 end points randomly, make the measurement, geolocate by using our three-stage framework finally. Table. IV shows the results of our five attempts. There were seven tests in total where better results could be achieved using the Looking Glass, which illustrates the potential of using the LG for IP geolocation problem. However, using LG does not always ensure that better results will be achieved. Our goal is not to replace the probe measurement but to provide more choices for researchers.

B. Router address vs Terminal address

In our study, we collected two distinct types of end points for IP geolocation, which were combined to evaluate in other experiments. However, we recognized that different network conditions may impact the accuracy of our method. To further investigate this, we trained our model using the same set of 32,000 end points to geolocate two different types of IPs: 3,000 terminal IPs and 3,000 router IPs. Our findings revealed that we were able to geolocate router IPs with an average error of 100 km, while the average error for terminal IPs was 800 km. These results support our hypothesis that a more stable network environment leads to more accurate geolocation predictions.

TABLE V: Average Error (km) using different algorithms

Number \ Algorithm	RF	DT	KNN	NN
2500	226.59	295.52	261.98	623.45
10000	128.69	202.30	152.42	324.26
17500	89.81	184.81	107.17	266.47
25000	81.82	166.06	97.70	230.84
32500	75.88	161.30	90.73	220.86
40000	71.72	148.91	85.97	212.46

C. Random Forest vs other algorithms

We choose the Random Forest to geolocate in stage 2 and stage 3. Now we compare its accuracy with other algorithms in IP geolocation problem. Because this problem is a multi-output regression problem, we select three other common machine learning algorithms for this kind of problem: decision trees (DT), k-nearest neighbors algorithm (KNN), and neural network (NN). Notably, we use two-tier fully connected neural network and the size of neurons in the hidden layer is twice as much as input size. Table V shows the accuracy of different algorithms. Among these machine learning algorithms, using the Random Forest achieves the greatest accuracy in our addresses. It validates our hypothesis that the RF is more suitable for handling noise-contained RTT data.