GeoCorrect: Automated method to update geospatial data using Volunteered Geographic Information

Suraj Nath Sidh, Prosenjit Gupta
surajnath.sidh@st.niituniversity.in, prosenjit.gupta@niituniversity.in
Department of Computer Science
NIIT University, Neemrana

Abstract
In this day and age of big data, data is growing exponentially and so is big spatial data. With the big spatial data comes the big challenge to keep it up to date, If we fail to keep our spatial data updated. It can result in less accurate inferences we derive from that data.
In this paper, we present an automated method to update spatial data using Volunteered Geographic Information(VGI) data. To demonstrate our method we have built an end-to-end system using check-in data of foursquare users which are used to improve the accuracy of the location of stores.

Introduction
Big Spatial data plays an important role in our daily life and if we look around we can see it in action in Google Maps, Wyze. It helps us move around and find places that we never knew about on a tap of a finger. But what happens when the underlying data of those systems are not up to date or inaccurate, We receive wrong route that says we may have to drive through a building to get to a destination or we may go to a place which has relocated or closed down, since our underlying data that was out of date with real information, we derived wrong inference and results from it.

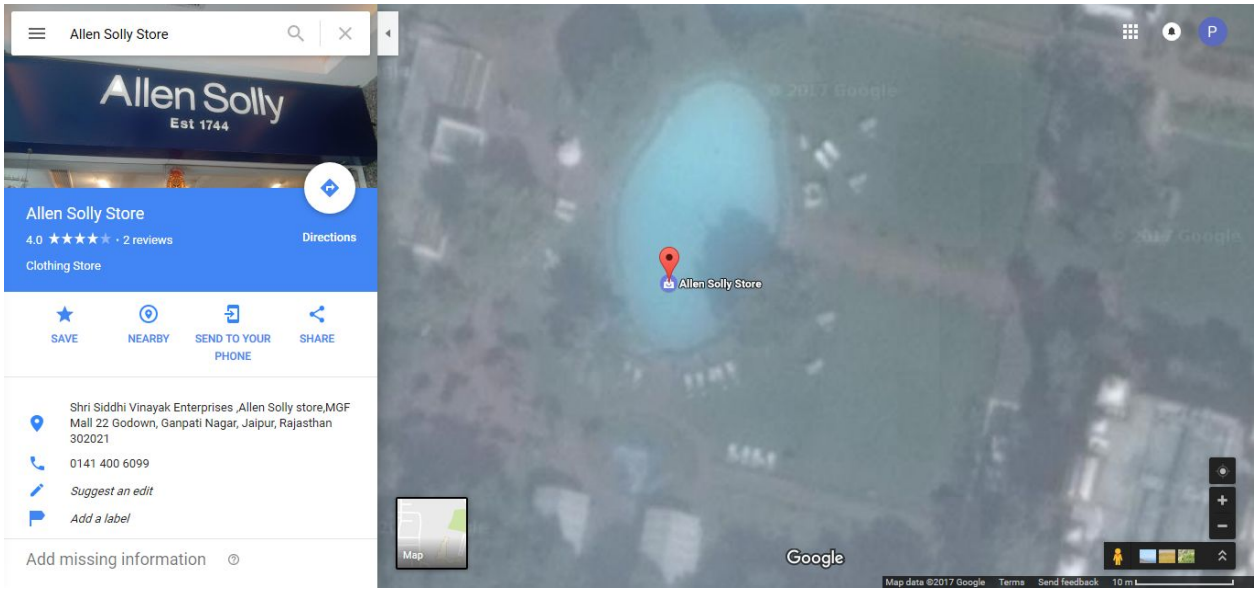


Image: an incorrectly marked geo-location on Google Maps

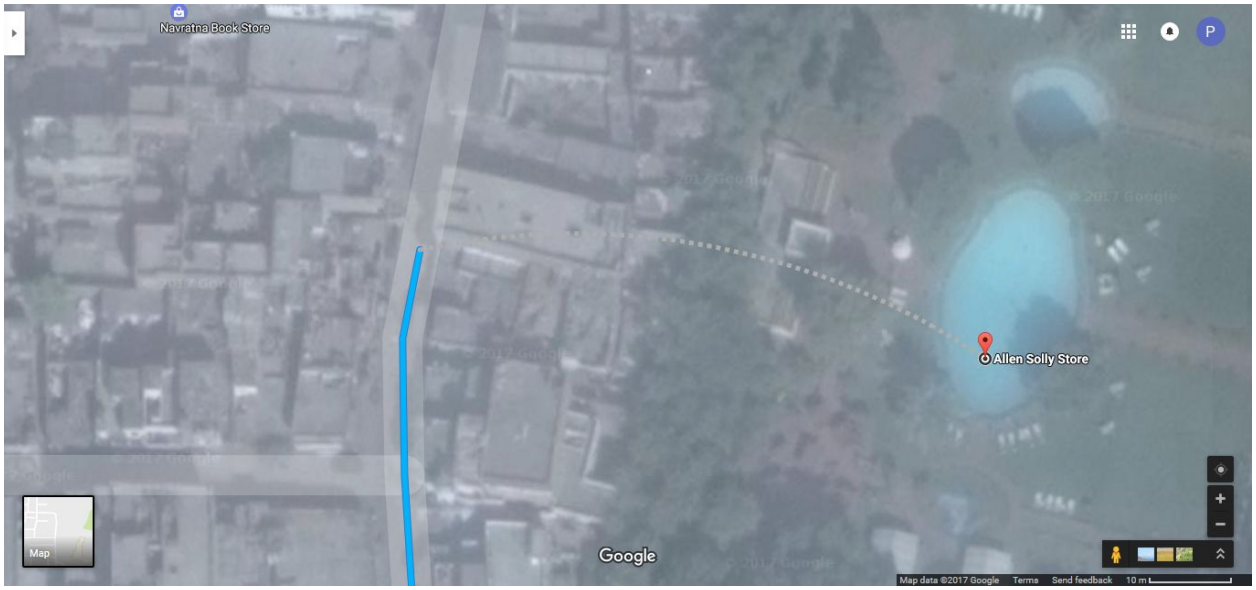


Image: Google Maps trying to show the route to an incorrectly marked geo-location

So how can we improve our systems and keep them up to date with real information, one of the old ways is to use authoritative data to update our geospatial data, maybe merge multiple authoritative data sources to cover more areas and improve our services but authoritative date maybe more updated than our databases but certainly it is not the most up to date source since authoritative survey takes time to complete by the time we get the data it is already out of date with real information, along with that we have to take care of merging multiple authoritative data sources which in itself is a huge problem since all GIS data does not follow the same structure and even much bigger problem is in case of contradiction which data source we should trust.

In this paper, we propose a crowdsourced approach to keep our big spatial data updated using Volunteered Geographic Information(VGI) data. In the past there have been many attempts to showcase that we can trust VGI data but it is vulnerable to group attacks(when a group of people decides to deliberately feed wrong information to system), but in those cases we were relying on humans to manually provide information which was not an efficient way even if we assume that people are ready to take time put in their data, which is clearly not the case
We propose a more effective way to collect VGI data where users just have to opt-in and verify or they voluntary check-in at places because it serves a purpose for them(for example Facebook/Foursquare Check-in that informs friends about their whereabouts). We can utilize that information to update underlying geospatial data of places they checked-in.

As we know commodity location devices are getting better and more accurate over time, nowadays GPS in our smartphones is more accurate and most modern smartphones use an ensemble of GPS and cell-tower information to provide location information. Which means now VGI data less error-prone and have better location resolution.

Related work

In the past in order to update geospatial data, we had to depend on authoritative data, so in past Majority of work in this domain mostly focuses on Matching, merging and integration of multiple geospatial dataset/Integration of geospatial information.

There has been work that showed us how to use VGI data along with authoritative data to build road networks.

Other then that we also have seen the effectiveness of VGI data, past work has shown how we can use VGI in disaster management, Building maps of dynamic areas for accessibility.

Dataset

Our goal was to use VGI information to correct location but we were not able to find any dataset that meets our requirements, so we used the dataset from Yang e. Al. which is a long-term (about 10 months) check-in data in New York City and Tokyo collected from Foursquare from 12 April 2012 to 16 February 2013 and extended this dataset to include ground truth location of places using the foursquare id of place(check-ins had foursquare place id as key identifier) and Foursquare API. location information obtained from Foursquare was used as ground truth location of a place. For our use case, we enriched foursquare check-in dataset to include ground truth location of stores, our system has used foursquare VGI data which then was preprocessed beforehand to remove outliers

Now our dataset contains check-ins(from Yang et al.) and ground truth location of a place(from Foursquare API)

Our dataset has the following schema

check-ins table have *venue_id, lat, lng* here *venue_id* is unique foursquare venue id, lat is the latitude of check-in, lng is the longitude of check-in

venues table have *venue_id, lat, lng* here *venue_id* is unique foursquare venue id, lat is the latitude of venue, lng is longitude of venue(here lat & lng are the location of venue as per Foursquare) Our dataset has 783511 check-ins for 97395 unique venues

Method

We are using mode to calculate corrected location because we have not removed outliers from our dataset, in an ideal scenario we don't have outliers but in real life, we can have some outliers due to many reasons which are faulty GPS sensor, floating point error in calculations and much more.

Method of calculation

- select a venue_id from venues table in dataset(here venue_id is unique)
- select all the check-in points for that venue_id from check-ins table
- find the mode of all the check-in points for a given venue_id by
    - find the mode of the latitude of all the check-in points for a venue_id
    - find the mode of longitude of all the check-in points for a venue_id
- the resulting location will be the new corrected location
    - mode of resulting latitude will give us resulting latitude
    - mode of resulting longitude will give us resulting longitude

As we can see we have a very simple method for calculating correct location which has some very simple assumptions along with it.
First assumptions are that our check-ins points are in a very close range, so we don't have to take the curvature of the earth in the account(we are assuming that we are working with a small piece of earth and that very small portion can be treated as a flat plan)
This assumption makes calculation simple and fast(speed is important when you have millions of venues with hundreds of check-in for each venue) and doesn't have any significant error penalty because at very small distance ranges this approximation is valid
Other assumptions are that we don't have any big outliers(way off from most of the points) in our data(as you can see, our method does not perform any outlier removal)

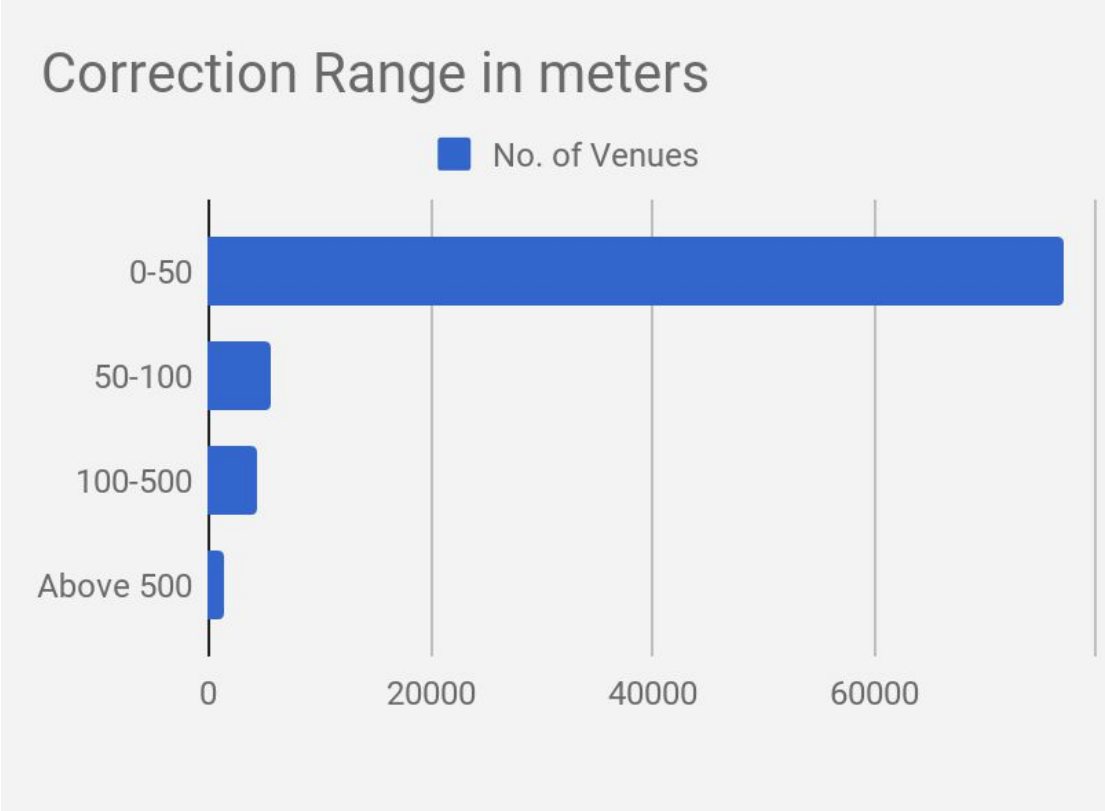
There are three cases for calculation of mode

1. venue_id doesn't exist in check-ins table.
    a. If there are no check-in points for an id
    b. present location in venues table is final.
2. venue_id exists and mode can be calculated.
    a. If there are check-in points and mode can be calculated
    b. calculated location is final.
3. venue_id exists but multiple modes exist for it.
    a. If multiple modes are found for a venue_id
    b. the location closer to the current location is treated as final.

Experiments
We implemented our method using python and ran it on our dataset (which have 783511 check-ins for 97395 unique venues) You can find our implementation and dataset on GitHub(https://github.com/electron0zero/GeoCorrect) In our dataset, we had an average change in distance(correction) of 258 meters on the total of 97395 venues. Out of which 9070 venues had no change at all and 88325 have shown change(correction), out of 88325 locations which showed some change, 77131 showed a change of 0-50 meters, 5591 showed change of 50-100 meters, 4268 showed change of 100-500 meters and finally 1335 venues showed change of greater than 500 meters.

Range(meters)	No. of Venues
0-50	77131
50-100	5591
100-500	4268
Above 500	1335

Table: range of change in venues



For calculation of distance, we used the location which we obtained from Foursquare using Foursquare API on (date of location data retrieval) and (date range when check-ins were obtained)

Visualization

We have built a visualization dashboard using data and results we obtained using our method using that data.
We used Google Maps API, Foursquare API, and our dataset and placed it on Google Maps and Google Street View, our visualization dashboard is built using Flask, Bootstrap, Google Maps API, and Foursquare API. it is currently hosted on Heroku(https://geocorrect.herokuapp.com/)

Conclusion

From our analysis we conclude that crowdsourced data (VGI) data can be used to perform map correction and updating, VGI Methods can be used to keep the Big spatial data up to date.
We suggest the use of gamification or reward system that motivates users to participate and share data.

Our system can be targeted by a group of people, deliberately feeding wrong information to system which can result in more error in underlying data(introduced by this attack) but this can be stopped using anomaly detection methods which will detect this sudden requests of update

for a certain location and a human intervention can make the final decision about those update requests.

Future Work

We believe results can be better if have more and better quality data. Data collected with accuracy/margin of error along with location information, where accuracy can be used as influence/voting power of a check-in in the final decision, more work on data collection, anomalous group attack detection and elimination would be a great addition.
Geospatial correction using VGI can be extended to indoor mapping and other mapping areas as well where VGI data is used to keep the spatial information up to date.
We can put multiple checkpoints in place to verify the update, it can range from asking the location owner to adding a report feature so users can report the mislabeled location.
Outlier detection and removal need more work, where the most trivial form of outlier detection can be done using geo-fencing as a venue and simply rejecting the check-ins received out of the geo-fence.

References

H. Qin, R. M. Rice, S. Fuhrmann, M. T. Rice, K. M. Curtin, and E. Ong, “Geocrowdsourcing and accessibility for dynamic environments,” *GeoJournal*, vol. 81, no. 5, pp. 699–716, Mar. 2015.

E. M. A. Xavier, F. J. Ariza-López, and M. A. Ureña-Cámara, “A Survey of Measures and Methods for Matching Geospatial Vector Datasets,” *ACM Computing Surveys*, vol. 49, no. 2, pp. 1–34, 2016.

D. Yang, D. Zhang, V. W. Zheng, and Z. Yu, “Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 1, pp. 129–142, 2015.

M. Zook, M. Graham, T. Shelton, and S. Gorman, “Volunteered Geographic Information and Crowdsourcing Disaster Relief: A Case Study of the Haitian Earthquake,” *SSRN Electronic Journal*, 2010.

Google Maps Embed API | Google Developers. [Online]. Available: https://developers.google.com/maps/documentation/embed/.

Flask-GoogleMaps 0.2.5 : Python Package Index. [Online]. Available: https://pypi.python.org/pypi/Flask-GoogleMaps/.

Foursquare, “Build To Inspire,” *Documentation - Foursquare Developer*. [Online]. Available: https://developer.foursquare.com/docs/.

Geopy, “geopy/geopy,” *GitHub*, 12-Sep-2016. [Online]. Available: https://github.com/geopy/geopy.

Mbr, “mbr/flask-bootstrap,” *GitHub*, 05-May-2017. [Online]. Available: https://github.com/mbr/flask-bootstrap.

“Understanding Latitude and Longitude,” *Annenberg Learner*. [Online]. Available: http://www.learner.org/jnorth/tm/LongitudeIntro.html.

“Welcome,” *Welcome | Flask (A Python Microframework)*. [Online]. Available:

http://flask.pocoo.org/.