Abstract

Identifying authoritative influencers related to a geographic area (geo-influencers) can aid content recommendation systems and local expert finding. This thesis addresses this important problem using Twitter data.

A geo-influencer is identified via the locations of its followers. On Twitter, due to privacy reasons, the location reported by followers is limited to profile via a textual string or messages with coordinates. However, this textual string is often not possible to geocode and less than 1% of message traffic provides coordinates. First, the error rates associated with Google's geocoder are studied and a classifier is built that gives a warning for self-reported locations that are likely incorrect. Second, it is shown that city-level geo-influencers can be identified without geocoding by leveraging the power of Google search and follower-followee network structure. Third, we illustrate that the global vs. local influencer, at the timezone level, can be identified using a classifier using the temporal features of the followers. For global influencers, spatiotemporal analysis helps understand the evolution of their popularity over time. When applied over message traffic, the approach can differentiate top trending topics and persons in different geographical regions. Fourth, we constrain a timezone to a set of possible countries and use language features for training a high-level geocoder to further localize an influencer's geographic area. Finally, we provide a repository of geo-influencers for applications related to content recommendation. The repository can be used for filtering influencers based on their audience's demographics related to location, time, language, gender, and ethnicity.

INFERRING DEGREE OF LOCALIZATION OF TWITTER PERSONS AND TOPICS THROUGH TIME, LANGUAGE, AND LOCATION FEATURES

by

Aleksey Valeriy Panasyuk

B.S. in Applied Mathematics, SUNY Polytechnic Institute, 2007

B.S. in Computer Science, SUNY Polytechnic Institute, 2007

M.S. in Computer Science, SUNY Polytechnic Institute, 2011

DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer & Information Science & Engineering

Syracuse University

May 2021

DISCLAIMER

The views and conclusions contained herein are those of the author and should not be interpreted as representing the official policies or endorsements, either expressed or implied, of the United States Air Force, Department of Defense, or the U.S. Government.

Acknowledgments

The purpose of education is to replace an empty mind with an open one.

- Malcolm Forbes

There were many people without whom this journey would have not been possible.

Dr. Yu, through a class, introduced me to social network analysis. Dr. Mehrotra was my initial advisor and even in his retirement continued being the most active. Dr. Mohan is whom I first talked to about the Ph.D. program, who introduced me to Dr. Mehrotra, and who guided me in the end. There was a lot of patience required. Looking back, I realize how little I knew. Looking forward, I realize how much I still have to learn.

To the other members of my defense committee, Dr. Ashok Sangani, Dr. Sucheta Soundarajan, Dr. Reza Zafarani, thank you for taking time out of your busy schedules to read this dissertation, provide valuable feedback and serve on my committee.

To the Information Directorate of the Air Force Research Laboratory and my supervisors, Dan Daskiewich and Doug Wynne, for supporting me in this endeavor. There were also many great mentors, I had along my professional journey, including Sheila Rakowsi, Mark Pronobis, Dr. Erik Blasch, and Dr. E. Paul Ratazzi.

To my parents who taught me to dream and not give up.

To my wife Yelena and children Natasha, Kristina, Andrey, and Alina for your love and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Consider that there is a crisis in some portion of the world. Using social media, the problem is to identify relevant influencers, what these influencers are saying, and the reactions from the ordinary populace. Examples from the past include the 2014 annexation of Crimea by Russia, the 2011 Arab Spring, and others. To solve the problem it is necessary to characterize the influencer's followers' geographic spread. More broadly, this analysis is important for (i) content recommendation, (ii) local expert finding, and (iii) location-aware influence maximization.

This research utilizes the Twitter platform. As an illustration, here are some of the more popular social media platforms based on the number of scholarly articles from Google Scholar: (i) Twitter 7.59M, (ii) Facebook: 6.69M, (iii) Reddit 1.74M, (iv) Instagram 1.63M, (v) Foursquare 0.048M.

Twitter is an ideal platform for fast-spreading news. Twitter users are forced to be brief with a message character limit of 280 characters. In comparison, on Reddit users are engaged in longer conversations and thus have more data for natural language processing applications such as sentiment analysis. While other platforms, such as Facebook, have safeguards that do not allow connecting to a user without a user's permission; Twitter is set up as a broadcasting platform where one user can quickly subscribe to any other.

1.1 Terminology

- Twitter user = can be an influencer or a follower.
- Influencer = a user with followers; generally at least 500 followers.
- Geo-Influencer = influencer whose followers are concentrated in a geographic area. The size of the geographic area depends on the feature used to quantify the localization. Using time-based features geographic area is at the timezone. Using location-based features the geographic area could be at the city-level.
- Follower = a user that displayed an interest in an influencer through a follow.
- Ordinary Follower = one who has between 20 and 100 friends, less than 500 followers, not verified by Twitter, without a URL, and that does not generate over five tweets per day since created.
- City-Community = made up of ordinary followers that reside in the city. Verified via Tweet-based Home Location (THL), Self-reported Home Location (SHL), or connection to a known city-level geo-influencer.

1.2 Twitter Challenges

Twitter users generate around 500 million daily tweets. Twitter has an Application Programming Interface (API) that allows collecting a portion of this data. The API has limits on number of calls allowed. Twitter does not allow researchers to share the collected data beyond the unique message and profile identifiers (others would have to collect on these ids to get the full dataset).

Most users are mindful of their privacy and as a result, try to report little or no personal information. If personal information is reported, it is done at a very high level, such as by reporting a textual string that may or may not contain their actual name and city-level location. A lot of these users are passive readers that do not generate message traffic (about one-fourth of all users have never posted a message).

If a Twitter user is known, their messages, followers, and friends can be collected. The Twitter API limits are such that it is not possible to collect all of Twitter by querying each user separately[1]. As a result, the social network graph is usually inferred from message traffic. If a user A, that generated the message, mentions user B, form a link between A and B. Around 1% of tweets can be collected using the free API.

Some messages contain coordinates but they are usually less than 1% of the Twitter stream. A more popular option is to utilize the textual self-reported location of the user that generated the message. The self-reported location needs to be geocoded. About two-thirds of all Twitter users are English speakers and for this reason, the geocoder and scenario usually revolve around English speakers. If the scenario involves a non-English speaking country it may be hard to find a good geocoder for that particular language.

Due to these challenges, a lot of works report collecting vasts amounts of data over many years to obtain a meaningful social graph. Furthermore, a lot of the message traffic is coming from bots (legitimate automated accounts) while around

[1]The current rate limit for "GET followers/ids" is 1 request per minute. There are over 300 million Twitter users which would equate to a collection lasting over 300 million minutes.

one-fourth of all Twitter users have never posted a single message. There is thus a danger in that the social graph may be more of a representation of what the bots are discussing vs. the ordinary populace. One may remove those that generate more than n daily messages, but the issue remains in that a lot of silent consumers and those that could not be geocoded will be lost.

1.3 Thesis and Contributions

In this research, we illustrate how targetted collection can be performed to identify influencers based on a geographic area of interest. The influencer's followers are studied for characterizing the influencer and for forming communities representative of ordinary users from the geographic area. The benefits are that the collection can occur faster and can capture a lot of users that are silent or hard to geocode.

On Twitter, when a user x follows a user y, it means that the user x will receive an update from Twitter whenever y posts a message. In a way, each follower is casting a vote for a particular influencer. The popularity of a user is a measure of how many others this user can reach and potentially influence.

When many followers are aggregated they can collectively provide useful information about the influencer. There are features, such as p_1% have a self-reported location that can be geocoded to 'New York NY' (related to location), p_2% of followers speak English (related to language), and there are also subtle features such as p_3% of followers whose creation time is during a specific hour h (related to time).

While the influencer's followers can help characterize the influencer, the influencer can in turn be used to characterize the followers. This is important because many followers will lack location information. Those followers that do not have location information, can be assigned a location based on other followers. This method will work, provided that the influencer is serving a small geographic area (a geo-influencer). For example, local traffic, local businesses, local news, and others are examples of such influencers. Geo-influencers cater their content to a specific geographic area and as a result, their followers tend to consist of users that are from or near the same geographic area. Following a local police department and local traffic serves as a strong indicator of the user's location even if the user does not list a valid self-reported location. Vice versa, an influencer with a global-like reach, with followers scattered globally, is not going to be useful for this task. Therefore, it is important to differentiate an influencer with a global following vs. a more local geo-influencer.

For characterizing the influencer we consider (i) location, (ii) language, and (iii) time-based features. An important novel finding of our research, is that (i) the influencer's follower localization can be characterized using only their temporal features and (ii) targetted collection leveraging Google search can identify geo-influencers without geocoding.

Our research has collected and utilized Twitter, but is applicable to social media in general. Social media typically involve users that can publish content and where the users can follow each other (followers and followee relations). As a result, if an approach is developed and works over one social network, it should work over

another social network provided that the same variables exist. Typically, every social network has a timestamp for when users post and when users create their accounts. For this reason, even though we did not collect data from Reddit, Facebook, etc. we believe that the approach is applicable to these other social networks (because their structure, their user base, and their features are similar enough to the ones explored in our research).

Chapter 2 analyzes geocoding performance on users from the USA. Questionable locations are identified by comparing the coordinates from user's messages to the geocoded coordinates from the self-reported location. Our research illustrates that Google's Geocoder is not well suited for data in the domain of Twitter. We show how additional parameters from the geocoder can be used to identify improperly geocoded locations and improve the geocoder. This helps identify 35% of locations that are improperly geocoded, mostly due to noisy locations being matched to a street-level address.

In Chapter 3, we propose a method for identifying city-level communities and associated geo-influencers. The method utilizes automated Google search queries to get a set of initial city-level geo-influencers. The followers of these geo-influencers are used for forming city-level communities. The follow connections from communities are utilized for identifying additional geo-influencers. Geo-influencers can be separated from more national types by analyzing if the influencer is connected to a single city-level community vs. multiple city-level communities. TF-IDF model where the terms are influencers can be used to generate a ranked list of influencers for each city-level community. The method is tested on 64 cities from the USA.

Chapter 4 proposes several ways for computing central location from influencer's followers. This central location is used to evaluate the method in Chapter 3. A means by which a repository of geo-influencers can be established is proposed. Chapter 4 illustrates queries that are problematic for Google. Other factors such as the follower sample size and query result order are examined. The chapter also illustrates that the followers of multiple geo-influencers are better aligned to the city, and hence result in a better city-level community.

In Chapter 5, we show how temporal features can be used for inferring the degree of localization for both social media users and message traffic. When applied over message traffic, the approach can differentiate top trending topics and persons in different geographical regions. Our analysis can help discover whether (and where) an influencer's followers are localized, even in the absence of geospatial tags. We demonstrate how several temporal features can be utilized for distinguishing local vs. global influencers. For global influencers, spatiotemporal analysis helps understand the evolution of their popularity over time. We can also infer the number of followers that were gained in a specified period, which assists in estimating link creation times. Thus, temporal features can assist in deducing and utilizing information about the numbers and locations of influencers' followers.

In Chapter 6, location information is incorporated for building a geocoding solution off of country region labels from temporal and language data. In the first step, of the proposed three-step approach, influencer's followers' creation times are used to create a time distribution to predict the time zone's Coordinated Universal Time (UTC) offset. The influencer is skipped if a UTC cannot be predicted with high

confidence. In the second step, the followers' language features are used to constrain set of countries associated with the UTC offset. After all influencers are processed and associated with regions, in the third step, a modified TF-IDF model is trained on region labels and associated followers' self-reported locations. The TF-IDF model learns popular ways that Twitter users refer to locations within the region in their native language. The multilingual TF-IDF model can then be used to infer the region of influence, for a new influencer, from influencer's followers locations.

Using the methods proposed in this paper, a repository of geo-influencers can be maintained. Chapter 7 shows an application related to content recommendation, whereby influencer is recommended using (i) the geographic locations, (ii) language, (iii) gender and (iv) ethnicity. A visualization, utilizing all of the main features proposed in the thesis, is presented, based on Kibana and ElasticSearch.

Chapter 2

Customizing an existing Geocoder for the Twitter Domain

Address geocoding, or simply geocoding, is the process of converting a human-readable, text-based description of a location, into a pair of (latitude, longitude) coordinates. Human language is complex, which makes geocoding service a non-trivial task. Geocoding is typically done via a public search engine (API service) through Google, Bing, Yahoo, and others. Geocoding services are designed to handle precise street-level addresses and their performance is, typically, tested for street-level addresses [18]. In this chapter we explore how well the Google's geocoder performs in the domain of Twitter.

We show that Google's geocoder makes mistakes due to the domain difference between social media and search engine queries. On social media, users are mindful of their privacy and hence are not likely to disclose their exact address. In contrast, search engine queries are not visible to the world and precise address is desired. Consequently, search engine users, searching for a business name, try to be as precise as possible.

Publicly available search engines, in particular, Google's geocoder, are prone to make mistakes. For example, the query 'New York, New York' gets associated with coordinates for 'New York New York Casino' in Las Vegas, NV. This happens due to the reason that, in the context of a search engine query, 'New York, New York' must

have had a higher click-through rate when a casino comes up vs. New York City. Similarly, ambiguous queries such as 'nowhere', 'worldwide', and 'my house' produce coordinates to a matching street-level business address, which are mostly erroneous.

Due to lack of alternatives, researchers utilize Google's geocoder for processing self-reported textual locations on Twitter. In this chapter, we evaluate this geocoder for geographical inconsistencies and errors and how to avoid or minimize them.

Our approach utilizes users that have both: (i) coordinates in messages and (ii) a self-reported location that produces coordinates via Google's geocoder. For these users, it is possible to determine error based on the distance between the geocoded self-reported location and the coordinates from messages. The average and standard deviation of the error are used to get the expected range for those self-reported locations that geocode and don't geocode well. This allows us to generate labels automatically and utilize them as training data for a binary classifier. We need a classifier because there are many users without coordinates in messages and thus the error measure between geocoder and message coordinates cannot be always computed. The final classifier gives a warning for 35% of locations; under 20% of users with such locations are confirmed using message coordinates (illustrating that they do indeed have poor performance).

2.1 Related Research

Recent literature reviews have focused on identifying popular research topics using Twitter [1, 2]. Karami et al. [1] performed topic modeling over 18,849 unique abstracts published between 2006 to 2019. Utilizing Latent Dirichlet Allocation (LDA) they categorized research into the most popular research topics: sentiment analysis, social network analysis, big data mining, topic modeling, and content analysis. Disease surveillance, tourism, politics, disaster management are some of the topics they discovered that require understanding location and that make it into the top 40 topics discovered.

There are three types of Twitter-related locations: user home location, tweet location, and mentioned location [3]. Our focus is on home location which comes from the self-reported location field in the user's profile. This field can be available for more than a third of the underlying users [4]. Having it for a large sample of Twitter population, makes it suitable for multiple applications, for example analyzing population demographics [5], user's spatial proximity [6], election polls [7], and flu affected areas [8].

To be useful, each self-reported location needs to be converted to latitude and longitude. To convert the location information to coordinates, researches often rely on a single geocoding service such as Google [5, 7]. A combination of services, can give higher confidence, for example when all report (latitude and longitude) coordinates within a short distance of each other [4, 6]. Gazetteer solutions such as GeoNames [9, 10] and custom parsers, to match on the city and state names, are other options

[8, 11].

The issue with the self-reported location field is that it can have ambiguous or irrelevant information such as 'Planet Earth' [4, 16]. Basic preprocessing, often via hand curated dictionaries, involves removing locations that are (i) vague ('France') and (ii) ambiguous ('Earth') [5, 6]. More advanced preprocessing involves breaking the string into address components, fixing each component for misspelling, abbreviation, and incorrect address format [12].

For validation of a geocoder, coordinates of the self-reported location are compared against the central location in user's messages. Typically users with messages that contain GPS coordinates are utilized. The coordinates are aggregated across a user's tweets where the most frequent city or the geometric median serves as the user's home location [3]. Researchers have reported that the user's home location from the self-reported field does not correlate well to the location inferred from tweets [4, 13]. It has been argued that the user-declared profile locations differ from the physical locations that are being tweeted from and hence cannot be used as useful proxies for the physical locations [4]. This is due to both – the self-reported locations having erroneous or incomplete information [13] and due to tweets that contain coordinates irrelevant to the user's home address [14].

It was also shown, that self-reported locations have a poor correlation with the expected population distribution. Researchers used self-reported locations as a Google-query, aggregated the returned location info by counties, and compared against 2000 US census data. Their findings showed that the Twitter population is a highly non-uniform sample of the population with mid-west underrepresented and more

populous counties over-represented [5].

However, there is evidence that the online communities correlate well with the network formed with geographic proximity. For example, researchers used network density and social distance to show that smaller networks are more socially clustered and extend a smaller physical distance [6]. A more recent paper illustrated that spatial proximity and geographic factors do affect online interactions [17].

In our study, we illustrate that some of the errors are due to the geocoder itself in that it attempts to match a home location not only on geographic name but also on establishment and business name. Hence, the discrepancy between self-reported locations and expected population distributions, observed in previous studies, may partially stem from these geocoding errors.

2.2 Data

Our data consists of 1,038,826 Twitter users with 131,925 unique nonempty self-reported locations. Each user had (i) a self-reported location that geocoder associated with the contiguous US and (ii) tweets contained either single point coordinate (coordinate geo-tag) or bounding box coordinates (place-tag). Place-tag is currently the default policy for associating geographic information with a tweet [15]. The bounding box coordinates from place-tag were transformed into a single point at the center. Point coordinates were available for 491,365 users.

For geocoding, we utilized a Python implementation for Google Maps Geocoding

V3 API[1]. Google allowed a maximum of 2500 queries per 24 hours so the geocoding results were collected over many weeks. Geocoder's region was set to 'US' and language to 'EN'.

2.3 Error Measure

We refer to the home location from coordinates in tweets as Tweet-based Home Location (THL) and the geocoded self-reported location in the user's profile as Self-reported Home Location (SHL). For each user its THL was computed using the geometric median over geo in user's tweets. The error for each user u is the distance in miles, computed using Vincenty's formula [19], between THL and SHL:

$$ED(u) = distance(THL(u), SHL(u)) \tag{2.1}$$

Three popular measures were used to measure the overall error across a group of users U [20]; the mean, median, and the proportion of users with error under 100 miles (note: higher values are better for $ACC@100$ while lower values are better for MeanED and MedianED):

$$MeanED = \frac{1}{|U|} \sum_{u \epsilon U} ED(u) \tag{2.2}$$

$$MedianED = median_{u \epsilon U}\{ED(u)\} \tag{2.3}$$

[1]https://developers.google.com/maps/documentation/geocoding

$$ACC@100 = \frac{|\{u \epsilon U | ED(u) \leq 100\}|}{|U|} \tag{2.4}$$

2.4 Categorizing Locations via Error Analysis

As a preprossing step we manually labeled 100 examples of geocodable and 50 examples of impossible to geocode strings. The impossible to geocode strings are mostly concepts that do not refer to an address, such as 'my house', 'the internet', 'everywhere', etc. Table 2.1 and 2.2 show sample of strings for each. The last row in each table shows the average and standard deviation across the three error measures.

The last row of Table 2.1, shows that $75 - 89\%$ (using average +/- 1 standard deviation of $ACC@100$) of users are within a 100-mile radius of geo from tweets. We checked that each maps to the expected city i.e. the geocoder provides accurate coordinates for these queries. Despite string being accurately geocoded, some of the coordinates from messages do not align because they may be from places visited that are far from the user's real home location.

Table 2.2 shows strings that are not possible to geocode since most refer to popular concepts instead of a physical address. From last row, we expect less than 3.5% (using average +/- 1 standard deviation of $ACC@100$) of users be within a 100-mile radius of geo in their tweets.

Looking at the $ACC@100$, the number of users, and expected error rates allowed us to look at specific locations more closely and put them into one of four categories. The expected best and expected worst error rates give the $ACC@100$ range for how

Table 2.1: Expected Error for Accurately Geocoded Strings (last row computed over 100 examples)

Location	**Users**	**MeanED**	**MedianED**	**ACC@100**
Los Angeles, CA	22431	436.21	9.89	0.79
New York, NY	15375	344.08	5.08	0.81
Chicago, IL	13788	229.53	6.11	0.81
Houston, TX	11676	186.95	7.09	0.83
Los Angeles	11380	302.02	9.89	0.86
Washington, DC	10500	300.96	1.48	0.8
...	...	...	...	...
Average±STD		205.1±.29	9.94±25.68	0.82±0.07

Table 2.2: Expected Error for Impossible to Geocode Strings (last row computed over 50 examples)

Location	**Users**	**MeanED**	**MedianED**	**ACC@100**
Earth	3125	3065.99	1308.67	0.00352
Worldwide	1277	2583.70	1231.46	0.00235
Everywhere	1042	1986.65	1251.54	0.02303
Global	850	2353.39	1229.36	0.03412
Planet Earth	722	2342.93	1544.00	0.0277
Hogwarts	574	3385.79	2141.28	0.01568
...	...	...	...	...
Average+/-STD		2657.75+/-1334.2	1968.74+/-1787.29	0.016+/-0.019

an accurately vs. inaccurately geocoded location is expected to behave. An example of each location category is shown in Table 2.3.

- Category 1: high $ACC@100$ (≥ 0.75 using lower bound for well-formed locations) and many users. Generally, it is a well-formed self-reported location that is accurately geocoded. Example: 'New York, NY'.

- Category 2: low $ACC@100$ (≤ 0.035 using upper bound of impossible to geocode locations) and many users. Generally, a popular human concept that is not possible to geocode such as Earth, Worldwide, and others.

- Category 3: average $ACC@100$ (near 0.5) and many users. Usually ambiguous that may be associated with multiple geographical places or a popular concept. For example New York, USA may refer to the state or the city. Disneyland is associated with the park in CA, but users may mention it as a popular concept without residing close to it.

- Category 4: low/high $ACC@100$ (near 0 or 1) and a few users. Given a small number of users it is hard to categorize individual locations as clearly right or wrong, i.e. we expect a poorly geocoded location to occasionally have high $ACC@100$ and vice versa.

Fig. 2.1 shows the ratio between Category 1 vs. Category 2 locations as the minimum number of users increases up to one-hundred. The figure illustrates that there are more training data points for Category 1. There are more category 1 locations with the ratio going from 10 to 1 for ten users to 20 to 1 for one-hundred

Table 2.3: Location Category Examples

C	Location to Google Mapping	Users	ACC@100
1	Los Angeles, CA to Los Angeles, CA, USA	22431	0.79484
2	Earth to 15612 S Keeler Terrace, Olathe, KS 66062, USA	3125	0.00352
3	New York, USA to New York, NY, USA	9010	0.57469
3	Disneyland to 1313 Disneyland Dr, Anaheim, CA 92802, USA	168	0.58333
4	Ocean Drive Miami, FL to Ocean Dr, Miami Beach, FL 33139, USA	1	0
4	Somewhere, Fishing to 1305 Snell Isle Blvd NE, St. Petersburg, FL 33704, USA	1	1

users. Locations associated with Category 1 and Category 2 will be used in the next section as training data for a classifier.

2.5 Classifier for Identifying Poor Geocoding

Google's geocoder will attempt to geocode queries such as 'my house' by matching to a business name. Such self-reported locations are common on Twitter due to the user being purposely ambiguous to preserve privacy. The classifier utilizes features, such as overlap between query and geocoder's associated address components, in order to make a prediction of whether Google's geocoder should be trusted.

Table 2.4: Google Geocoder Output

Additional Google Geocoder Output besides Coordinates

Output Type	**Description**
address components	Each component consists of the component type, short name, and long name. Example types are country, ADM1 (state), ADM2 (county), locality (city), street number, route, neighborhood name, postal code, and others.
formatted address	Full address matched by Google, may not include all of the address components (for example county usually omitted).
address type	Describes what the address is associated with, example values: point of interest, university, restaurant, and others.

Google Geocoder Output for query 'New York, New York'

Output Type	**Output Value**
coordinates	lat: 36.1023715, lng: -115.1745559
address components	street_number: 3790, route: South Las Vegas Boulevard locality: Las Vegas, ADM2: Clark County, ADM1: Nevada, country: United States, postal_code: 89109
formatted address	3790 S Las Vegas Blvd, Las Vegas, NV 89109, USA
address type	casino, establishment, lodging, point_of_interest

Table 2.5: Top 10 Most Frequent Address Type (Left) and Component (Right)

Address Type	**Ratio**	**Address Component**	**Ratio**
political	0.6298	country	1
locality	0.566	administrative_area_level_1	1
establishment	0.3101	locality	0.9639
point_of_interest	0.3068	administrative_area_level_2	0.9481
store	0.0563	postal_code	0.5652
food	0.0511	route	0.3348
neighborhood	0.04	street_number	0.3027
restaurant	0.0395	administrative_area_level_3	0.2005
university	0.0249	neighborhood	0.1827
route	0.0246	postal_code_suffix	0.1308

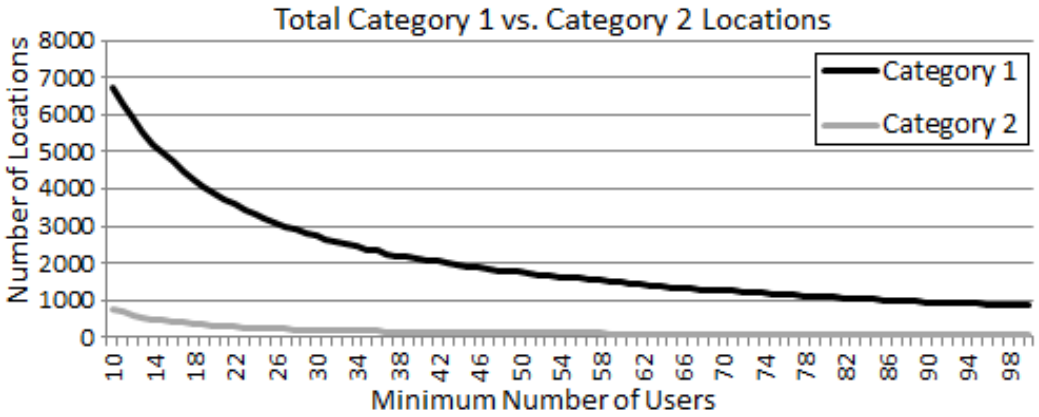


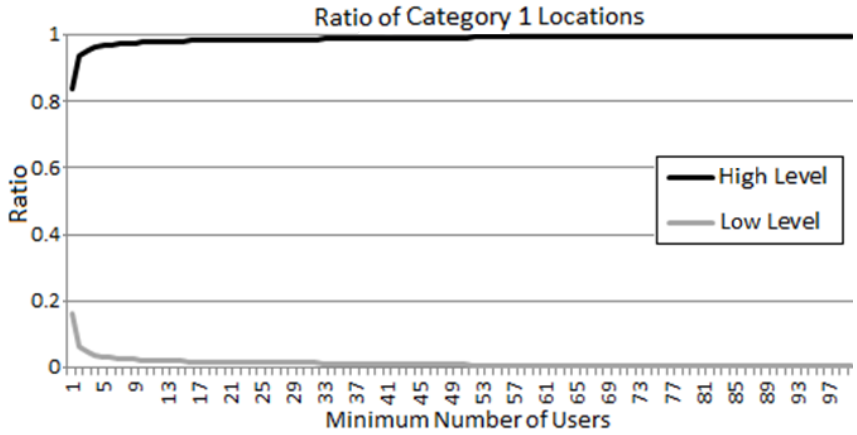
Figure 2.1: Category 1 are locations with a high accuracy where at least 75% of users post messages with coordinates that are within 100 miles of coordinates geocoded from the self-reported location. Category 2 has low accuracy where less than 3.4% of users match.

2.5.1 Features

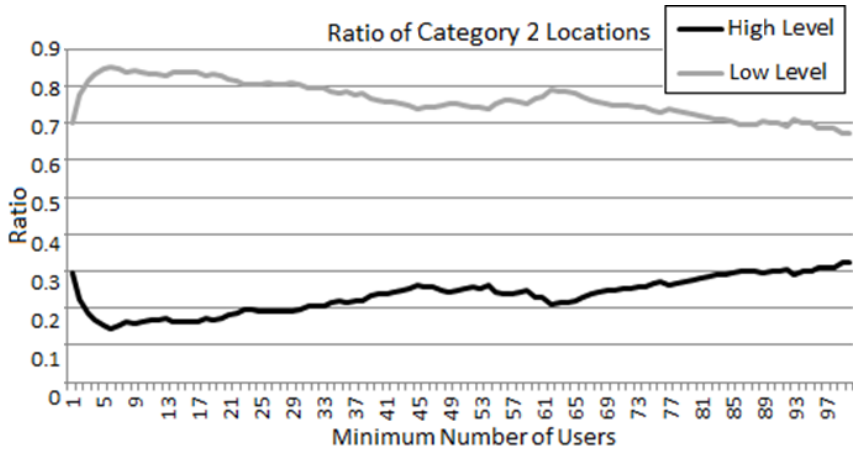
Table 2.4 shows additional output, Google's geocoder produces, besides the latitude and longitude coordinates, and gives as an example, the output for query 'New York, New York'. Notice that for the query, the street number in address components as well as the address types: (i) casino, (ii) establishment, (iii) lodging are indicative of an address that is a street level address (which could be used to assign a lower confidence for this prediction).

Across all of the locations in our dataset, there were a total of 73 unique address components and 116 unique address types. Table 2.5 shows the top 10 address components and address types that account for the biggest ratio of all locations. The address types of store, food, restaurant should not represent a plausible location that most Twitter users will associate themselves with, but surprisingly over 5% of locations in our dataset are some sort of a store (indicating potential errors).

After looking at all address components, our expectation was that most Twitter users will report a location that is associated with: (i) political entity, (ii) zip, or (iii) university address type (as these are large enough to be reasonable locations to associate with). A location with one or more of these attributes is classified as a high-level location; otherwise, it is classified as low-level (street-level).



(a) Category 1



(b) Category 2

Figure 2.2: **Top** – Large ratio of Category 1 locations associated with a high-level address (as in city-level). **Bottom** – Large ratio of Category 2 locations (impossible to geocode) are associated with low-level (as in street-level). The chart confirms that majority of properly geocoded locations are not at street-level.

Fig. 2.2 (top) shows that the majority of category 1 locations get associated with a high-level location type. This is especially true as the minimum number of users that utilize location increases. For locations with number of users ≥ 100, the high-level location type captures 99.6% (843 out of 846) of category 1 locations.

Conversely, Fig. 2.2 (bottom) shows inaccurately geocoded locations captured by category 2 are associated with low-level location type. The overall trend does not decrease, but the number of associated mistakes is small because the number of category 2 locations is small (only 43 locations used by at least 100 users). Examples of locations that do get associated with a high-level location type: Midwest to Midwest WY, Nederland to Nederland CO, Nowhere to Nowhere OK, Moon to Moon PA, and others where a popular concept matches a city name.

A number of additional features were proposed based on overlap between query and geocoder association. All of the features proposed are summarized below:

- F1: Political Entity = political address type without a street number address component. The political address type refers to recognized divisions of a physical territory; locality, neighborhood, colloquial area, sub-locality, and others.
- F2: Zip = postal code address type
- F3: University = university address type
- F4: Text Overlap = returns percent character overlap between textual self-reported location and textual address associated by the geocoder.
- F5: City/State exact = returns true if tokens from the query can be combined

to match city and state address component exactly, false otherwise.

- F6: Populous City = for unique cities with a population over 50K, it is assumed that city name may be known to most human users such that the state need not be spelled out. Location matched to 2016 US census data using city and state that Google associates with the query string.

2.5.2 Classifier

Accurately geocoded locations (TRUE label) are those with $ACC@100 \geq 0.75$ (category 1). Impossible to geocode locations (FALSE label) are those with $ACC@100 \leq 0.035$ (category 2) (each self-reported location used by at least fifty users). The classifier utilizes the proposed features for predicting when Google's geocoder will perform poorly.

For the classifier, we considered Naïve Bayes and Decision Tree (using gain ratio, information gain, and Chi-square interaction detector (CHAID)). Classifier trained using 5-fold-cross-validation utilizing the RapidMiner software package. Fig. 2.3 shows, the best performing classifier, Decision Tree using the CHAID criterion.

2.5.3 Performance

Table 2.6 compares the performance using three error measures proposed for locations that pass and fail classifier. Out of 131,925 self-reported locations in our dataset, 46,091 or 35% were classified as low confidence (Google geocoder's output should not

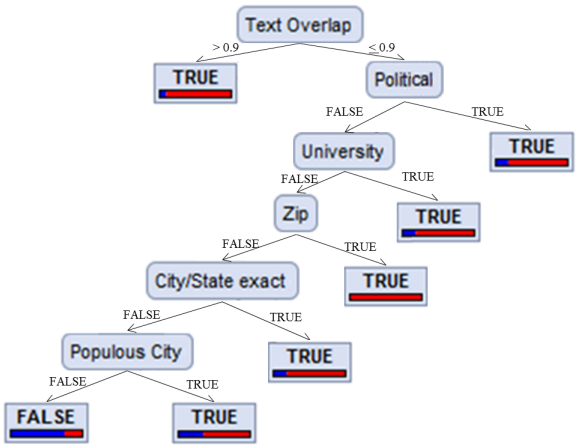


Figure 2.3: Decision Tree Classifier for Identifying Geocoding Errors

Table 2.6: Geocoding Performance over Users

Location Set	**MeanED**	**MedianED**	**ACC@100**
Fail Classifier	1705.12	879.45	0.1969
Pass Classifier	237.1	6.3	0.8027

be trusted for these). Under 20% of users with such locations are confirmed using message coordinates (illustrating that they do indeed have poor performance). In contrast, for those locations that pass the classifier, over 80% of users are confirmed via message coordinates.

The rules of the classifier illustrate that it is important to consider whether a location that is matched by the geocoder contains both the city and state as this is less ambiguous than a city name by itself. Google's geocoder is limited by the amount of API calls it can freely make daily and thus matching using a rule-based approach (using locations that contain a known city/state or city/country) is an option for a high precision/low recall solution.

2.6 Conclusions

The research has explored various types of geocoding errors and has established expected error rates for well and poorly geocoded locations in the context of Twitter. These measures were used to develop a classifier for whether the commercial off-the-shelf geocoder is performing as it should on Twitter data. In our dataset, close to 35% of self-reported locations geocoded using Google resulted in a warning. Under 20% of users with such locations were confirmed using message coordinates, illustrating that the Geocoder does exhibit a poor performance for these locations. In contrast, for those locations that pass the classifier, over 80% of users are confirmed via message coordinates. In the next chapters, for those users whose location cannot be determined from textual self-reported location, other features will be described for inferring location.

Chapter 3

Identifying Local Influencers and City-Level Communities

3.1 Background

User's popularity on Twitter can be measured by how well the user is recognized by others, such as through others' mentions and follows. Understanding the location of popular users is needed for content recommendation and other use cases. For example, an advertising agency, that is performing a city-wide promotion, will be interested in users that serve important roles within that city such as the city's mayor. Generating crime statistics across cities could require focusing on local fire, police, and other emergency related influencers. Tracking sports could require tracking local football, basketball, baseball, and others relevant to the city of interest.

The standard approach first geocodes a large number of users using each user's self-reported location. Second, the users whose locations map to within x miles of the city of interest are used to establish the city-level community. As has already been noted in Chapter 2, the biggest obstacle to this approach is the lack of a good geocoding solution.

In contrast, our approach does not require geocoding and instead leverages the power of Google search along with the follow structure on Twitter. We found that querying for (city, state, plus keyword 'Twitter') is likely to return Twitter influencers

that are specific to the city (geo-influencers). The geo-influencer's followers form the basis of a city-community. The benefit is that many of the followers may not contain a geocodable self-reported location, but make it in as part of our approach (this is because following a geo-influencer serves as a strong indicator of the follower's location). Following a local police department and local traffic updates serve as a strong indicator of the user's location even if the user does not list a geocodable self-reported location field.

The Term Frequency-Inverse Document Frequency (TF-IDF), where the Term Frequency measures the number of follows by the community, is used to produce a ranked list of most popular geo-influencers. In this way, by fusing the power of Google for finding initial geo-influencers and the crowd-sourcing power of the underlying Twitter community, we can associate hundreds of additional city-level geo-influencers and use these to further refine the city-community. A ranking of national-level influencers are those with followers across multiple city-level communities.

3.2 Related Research

There is a great amount of research related to Twitter user's activity, popularity, and influence [21]. Activity measures actions that a user takes (such as tweets, retweets, mentions, and replies); influence measures whether user's actions are capable of affecting other users' actions in the network; and popularity measures how well the user is recognized such as through others users' mentions and follows. In this research,

we are focusing on popularity through other user's follows. This work is related to Location-Aware Influence Maximization (LAIM) [22, 23], where the goal is to identify top k users to maximize the expected number of influenced users in a specific geographical area.

Multiple features can be used to establish a community of users with some features in common [24]. For example, (i) structure-based features where two users both follow the same influencer [25-28], (ii) activity-based features such as how frequently and during what times the user is active [29-31], (iii) content-based features such as the type of users, topics, URLs being mentioned [32-34], and (iv) communication-based features such as retweet, reply, and mention [35, 36].

The second step is to associate users with a geographical area (usually at the city-level). The user's location may be geocoded from the self-reported profile location or aggregated from coordinates in the user's tweets [3]. For a user without location information, the median of user's friends' location can be used [13, 37].

Finally, once the communities and the underlying geographical locations are known, the ranking of the users for each geographical area can be done using traditional graph metrics such as degree centrality [38], custom measures that for example punish spammers [39], and variations on PageRank [40-42].

A recent paper, that is most directly related to our research, explores the problem of identifying relevant geo-influencers across three US cities: Boston MA, Bristol CT and Seattle WA [42]. Their network was built using social activity based interactions

retweet, reply, and mention present in over five billion tweets. They relied on self-reported profile location information for extracting a ranked list of influential users whose location is within 100km of the city of interest. The ranking was performed via several modified PageRank based algorithms. They showed that self-reported locations were needed for filtering out global users that are not from the area such as @YouTube. However, it was also shown that limiting users within x miles of the location of interest would filter out other important users, such as @Patriots, that had a strong local connection spanning beyond 100 km.

We propose a novel way of performing a targeted collection that results in a community of users for each city location. For each city, the communities stem from an initial pool of influential users identified via automatic Google searches. Our approach does not rely on any location information. In this way, each community may contain passive readers that do not tweet and users with no location information. Most importantly our city communities are made up of followers of geo-influencers that are known to be associated with the city of interest. A modified TF-IDF measure results in ranked lists that perform well against hand-labeled data including data from [42], but the overall collection requirements are magnitudes smaller.

3.3 Approach

Given a city of interest and a list of other cities from which to distinguish, our approach will follow the process outlined in Fig. 3.1. The figure shows four main steps: (1) Google search to discover known geo-influencers for each city, (2) ordinary

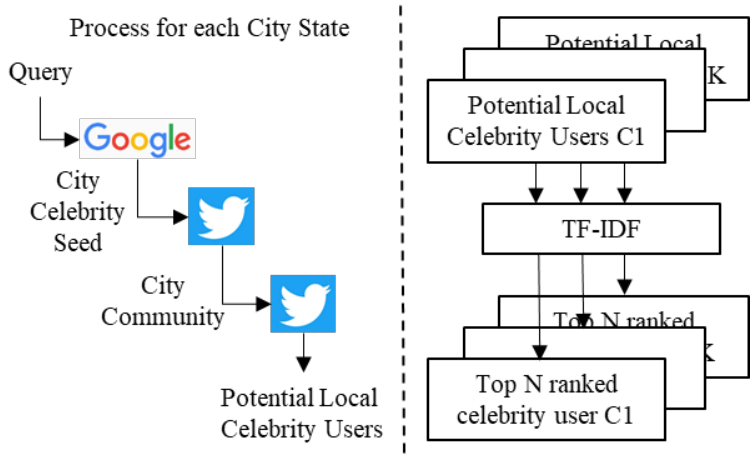


Figure 3.1: Top N geo-influencers extracted from K Cities

followers of geo-influencers from step 1 are used to establish a city community, (3) collect friends of the city community from step 2, and (4) a modified TF-IDF measure is used to identify geo-influencers from step 3. In our work, we define 'an ordinary follower' as one who has between 20 and 100 friends, less than 500 followers, not verified by Twitter, without a URL, and that does not generate over five tweets per day since created. An influencer is simply a user with at least 500 followers.

Steps 1-3 shown on the left side of Fig. 3.1 are repeated for each city after which step 4, illustrated on the right, is applied. Each component of the process from Fig. 3.1 is described in more detail in the subsections below.

Step 1: Google Queries for getting the Initial Geo-Influencers– Given a query that consists of (city, state, 'Twitter') (and an optional keyword such as 'Sports') Google returns a list of URLs. In the first 100 Google hits, our interest is in the following URL structure: 'https://twitter.com/' + screenname + '?lang=en'. We record the screenname and the associated URL hit number. In this manner,

Google search associates influencers with a city. In our work, the top ten influencers are used. By utilizing optional keywords, such as 'News' or 'Sports', we can focus on the geo-influencers by topic. For example, the query 'Syracuse, NY Twitter News' results in the top three news-related influencers: @SyracuseUNews, @syracusedotcom, and @NewsChannel9.

Step 2: Initial Geo-Influencers to City Community– Twitter API is used to collect followers of initial geo-influencers; thus forming the basis of the city community. According to a 2016 Twitter SEC filing, approximately 8.5% of all Twitter users are bots [43]. Bots have many connections and post many messages. Thus, to avoid/minimize bots we focus on ordinary followers (one of the criteria is not posting over five messages a day). To focus on the users that are interested in a single city, we also ensure that city-communities are disjoint, i.e., no user belongs to two or more communities.

Step 3: City Community to Additional Geo-Influencers– Twitter API is used to collect friends of users that make up the city community; users with over 500 followers form a pool of additional 'potential' geo-influencers; potential because influencers which are popular across multiple cities may be included (TF-IDF measure helps filter these out).

Step 4: Ranking via TF-IDF– As mentioned earlier, traditional approaches find the influencers using network based methods, such as the degree centrality or PageRank. In our research, we apply TF-IDF, which is intended to reflect how important a word is to a document in a corpus. To apply TF-IDF measure it was critical to have well defined city communities (those consisting of users that are from the city).

For each city, a document is made up of all potential geo-influencers that the city community has made a connection to. The process shown on the left in Fig. 3.1 builds such a document for each city.

Let C represent the set of all cities, community(c_v) return the set of ordinary followers that make up the community for city c_v, and friends(o_w) return the set of influencers that the user o_w follows:

- C = $\{c_1, c_2, ..., c_k\}$
- community(c_v) = $\{o_1, o_2, ..., o_m\}$
- friends(o_w) = $\{u_1, u_2, ..., u_n\}$

Term frequency for user u_x and community c_v corresponds to total friend connections to user u_x within community divided by total friend connections:

$$TF(u_x, c_v) = \frac{\sum_{o \epsilon community(c_v)} |u_x \epsilon friends(o)|}{\sum_{o \epsilon community(c_v)} |friends(o)|} \tag{3.1}$$

As an example, given community(c_1) = $\{o_1, o_2, o_3\}$ where friends(o_1) = $\{u_1, u_2\}$, friends(o_2) =$\{u_2, u_3\}$, and friends(o_3) = $\{u_1, u_2, u_3\}$: TF(u_1, c_1) =(1+0+1)/(2+2+3) = 2/7, similarly TF(u_2, c_1) = 3/7 and TF(u_3, c_1) = 2/7.

Inverse document frequency is given by the total number of cities divided by the number of cities user u_x has a connection to (cities where that user is mentioned more than once):

$$IDF(u_x) = \log\left(\frac{|C|}{\sum_{c_v \epsilon C} |\{TF(u_x, c_v) > 0\}|}\right) \tag{3.2}$$

As an example, given another community(c_2) = $\{o_4\}$ where friends(o_4) = $\{u_1, u_4\}$: IDF(u_1) = log(2/(1+1)) = 0 and IDF(u_4) = log(2/(0+1)) = log(2).

Combining formula 1 and 2 gives a formula for ranking a potential influencer u_x for community c_v:

$$TF - IDF(u_x, c_v) = TF(u_x, c_v) * IDF(u_x) \tag{3.3}$$

3.4 Data

Using the data from the 2016 Census Bureau we focused on known cities in the USA. We built a set of cities by initially starting with the most populous city and incrementally adding other most populous cities as long as they were at least 30 miles apart from cities already in the set. This ensured that the selected cities were geographically spread apart. The set so obtained contained 264 (city, state) pairs.

The process in Fig. 3.1 was used to generate three datasets based on three different keywords that made up the Google query: (i) Twitter, (ii) Twitter News, and (iii) Twitter Sports. Out of 264 cities, only 64 cities contained at least ten geo-influencers for each search type. Followers of geo-influencers were used to establish each city community. The followers that simultaneously follow many geo-influencers were prioritized. It was required that each city community have at least 100 and at most 1000 users. Fig. 3.2 shows the collection process for the network associated with each city and the corresponding maximum network size.

In addition to the three datasets described above, we obtained the fourth dataset

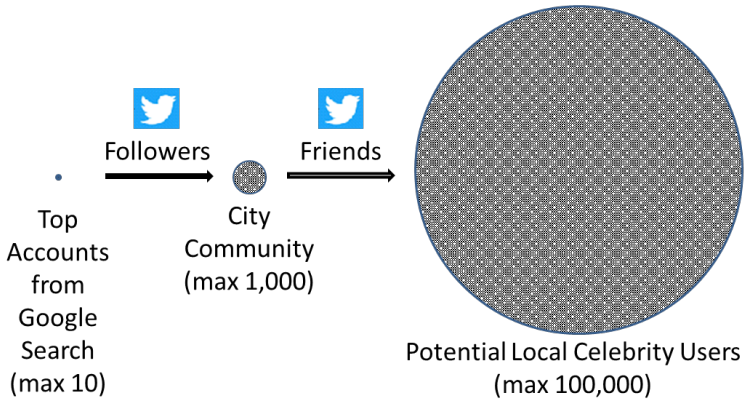


Figure 3.2: Collection process from initial influencers to city-level community to a new set of influencers (drawn to scale)

where the city community is made up of users whose self-reported locations were verified via geocoder (using Google Maps API[1]) to within 10 miles of city's (latitude, longitude). For this dataset, ordinary followers were selected from the followers of major news outlets: ABC, Politico, PBS, WSJ, Fox News, Reuters, CNN, and MSNBC. For each city community, we collected at least 100 users but at most 1/500 of the city's population. Out of the 64 cities covered by the other three datasets, only 19 cities provided a large enough sample size. These 19 city communities across the four datasets were: Charlotte NC, Washington DC, Wichita KS, Tucson AZ, Denver CO, Madison WI, San Diego CA, Syracuse NY, Lansing MI, Toledo OH, Boston MA, Columbia MO, Chicago IL, Springfield MO, Rochester NY, Lubbock TX, Atlanta GA, Topeka KS, and Albany NY.

Comparing results from the first three datasets showed the effect of differing queries, i.e., how the resulting communities differ in geo-influencer ranking. The

[1]https://developers.google.com/maps/documentation/geocoding

Table 3.1: Dataset Statistics across 19 Cities that occur within each dataset. The total number of users across the 19 city-communities is shown as well as the average number of users per city-community and the associated standard deviation.

Set	**Seed Type**	**Avg**	**Stnd**	**Total**
D1	Google using Twitter	692.21	203.52	13152
D2	Google using Twitter News	728.95	198.35	13850
D3	Google using Twitter Sports	516.32	324.30	9810
D4	Major National News	726.79	944.03	13809

fourth dataset allows us to compare how the city community generated via geo information differs from community based on follow connections to Google identified influencers. Table 3.1 summarizes the four datasets collected across these 19 cities. Table 3.1 shows the number of users, average, and standard deviation per community. From the table, we see that Dataset 4 has a high standard deviation, i.e., due to bigger samples for bigger cities. Dataset 3, related to 'Twitter Sports', has a smaller community than the more general topics: 'Twitter' and 'Twitter News'.

3.5 Evaluation

3.5.1 Impact of Keyword in Google query on Final Rank

We analyzed the first three datasets. The ground truth, in this case, are the geo-influencers using Google search. Each dataset contained 640 geo-influencers across 64 cities (1920 for all three datasets). Venn diagram in Fig. 3.3 (left) shows 79 out of 1920 or 4.1% of geo-influencers associated by Google overlap across three search

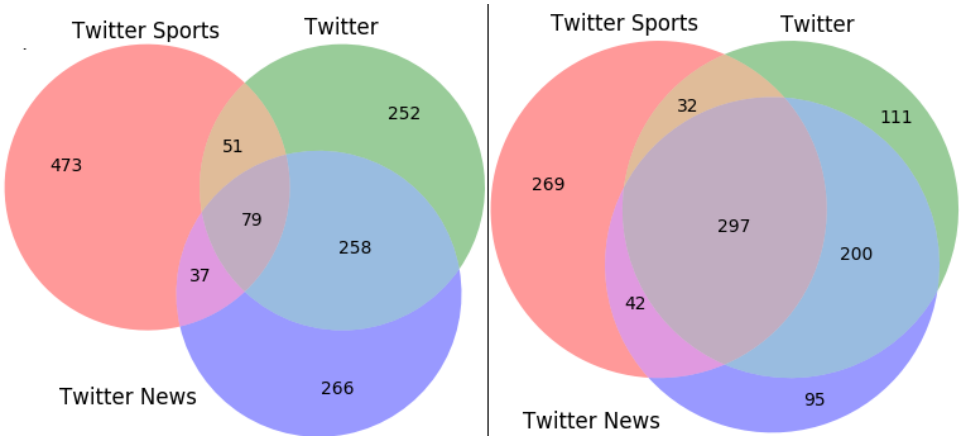


Figure 3.3: Seed Twitter User Overlap (left) vs. Top Ranked User Overlap (right) across 64 cities for the three search types). While the initial queries produce an initial set of influencers (seed) that differ (shown on the left), the resulting influencers that are extracted from city-level communities formed from seed have quite a bit of overlap (shown on the right). This illustrates that the method is robust in that different influencers can be used as a seed to get to the same end result (as long as the initial influencers are indeed local to the city of interest).

types with more overlap between Twitter and Twitter News. Ranked list via TF-IDF, Venn diagram of Fig. 3.3 (right) shows a more significant overlap, 297 out of 1914 or 15.5% of users. The figure shows that on average more than half of ranked geo-influencers overlap (with Twitter and Twitter News being most aligned). This illustrates that different initial geo-influencers can lead to similar final rankings.

3.5.2 Performance against Google Ranked Geo-Influencers

Google is driven by those webpages that a user is most likely to click on while the number of followers drives Twitter's popularity. In the majority of cases we have found that there is an overlap between Google associated users and top-ranked users via TF-IDF, especially as n increases. We calculated the ratio of overlap between

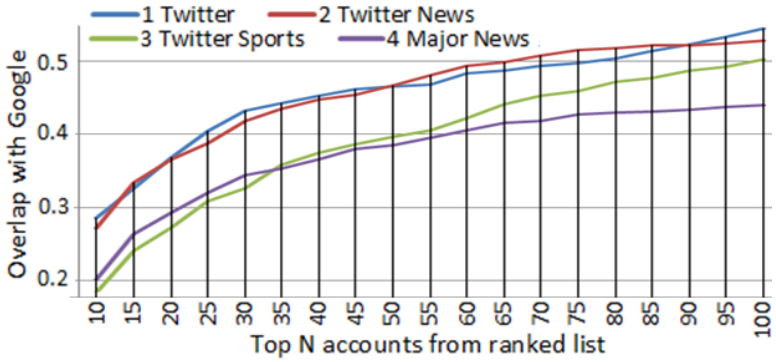


Figure 3.4: Percent of Google geo-influencers confirmed by each query type as the ranked list grows in size. Followers of major news that were verified via geocoder follow similar influencers as returned from Google.

the two as:

$$Overlap(X(c_v), Y(c_v, d, n)) = \frac{|X(c_v) \cap Y(c_v, d, n)|}{|X(c_v)|} \tag{3.4}$$

Where X(c_v) refers to a set of known geo-influencers that are associated with city c_v from Google and Y(c_v, d, n) refers to top n ranked influencers stemming from the TF-IDF measure for city c_v and dataset d.

The top ten geo-influencers across three Google search types are combined providing a larger set to compare against (24.58 geo-influencers on average). Fig. 3.4 shows, for each dataset, the average percent of Google geo-influencers confirmed by each query type as the ranked list grows in size. The metric is averaged across 19 cities that are present in each of the four datasets. Despite each dataset representing slightly differing communities, due to having been formed using different seeds, we see that they all confirm a similarly large portion of geo-influencers that were deemed relevant by Google with generic Twitter search type working the best.

Table 3.2: Overlap for Geo-Influencers from Google vs. Dataset

Search Type (X)	**D1**	**D2 News**	**D3 Sports**	**D4 News**
Twitter	**0.69**	0.62	0.37	0.55
Twitter News	0.71	**0.74**	0.47	0.61
Twitter Sports	0.12	0.12	**0.28**	0.08
all	**0.43**	0.42	0.33	0.34

In a similar fashion, Table 3.2 shows the top 30 geo-influencers using TF-IDF vs. top ten geo-influencers for each Google search type. The last row is a combination of the top ten Google geo-influencers across datasets D1-D3, also used in Fig. 3.4. Ratios in bold highlight the best aligning datasets. It could be argued that each dataset simply confirms the same initial seed that was used to establish the dataset. D4 was established, to illustrate that communities established without leveraging Google would also confirm them. Communities from D4 were established from self-reported locations of followers following major news influencers and hence did not utilize Google in any way. Despite this D4 had a high overlap for Twitter News search type (0.61, this overlap helps confirm that the proposed approach can work in a similar fashion as the one that relies on geocoding).

Next, we focus on Google geo-influencers that do not make it into ranked lists even for very large n. For instance, for n=1000 there were 67 such geo-influencers. We checked each of the 67 geo-influencers by hand. 37 out of 67 geo-influencers were found accurate, but all had fewer than 500 followers to be considered by our method. These referred to local high schools, local businesses, small local sports teams related to tennis, volleyball, and others. To incorporate these geo-influencers, we would need to change our threshold for influencer from 500 to 100 followers.

Next, examples of Google geo-influencers that were found inaccurate. @CityofToledo has never tweeted, with 105 followers, being in the fourth search result for the query 'Toledo OH Twitter' (this is probably due to close overlap in keywords with @city_of_toledo the first search result and official city account). Some influencers have a national follower base, but are associated with a single city such as @RiddellSports associated with 'Chicago, IL Twitter Sports'. Google search results fluctuate over time with about two-thirds of the 67 accounts no longer recommended after the search was repeated about a month later. Hence verification is still recommended for optimal results when establishing initial geo-influencers from which each city community is to be established.

3.5.3 Communities via Location vs. Google Seed

Dataset D4 was established using followers of major national news outlets whose self-reported location could be associated with the city of interest. The issue with self-reported locations is that less than a quarter of all users had locations that could be geocoded. Also, we found geocoding errors for example 'Salem' was associated with 'Salem, MA', but often the location referred to Salem which is in India. As a result, we focused on high confidence locations that specify both city and state, but such locations were challenging to find for smaller cities despite collecting on millions of users.

The benefit of datasets D1-D3 is that they were assembled much quicker than

D4 and they covered many more cities with more users per city. For users in D1-D3, over 64 city communities, the percent of users with the non-empty self-reported location was only 28.2%. For those users that did specify their location, on average 41.6% contained the city name associated by our method, thus illustrating that these communities are well structured.

3.5.4 City Level Evaluation

Evaluation in [42] listed ground truth for Boston MA via these 20 geo-influencers: (i) News = wcvb, bostondotcom, cbsboston, 7news, bostonherald, (ii) Sports = redsox, celtics, nhlbruins, thebostonpride, bostoncannons, (iii) Gov = marty_walsh, cityofboston, bostonpolice, bostonfire, masddot, and (iv) University = bu_tweets, harvard, mit, berkleecollege, northeastern (influencer cbsboston renamed to wbz). The top five for the best performing ranked list from [42] contained: Patriots, BostonGlobe, OnlyInBOS, RedSox, and NHLBruins (so in their approach the last two matched the ground truth). For our approach, table 3.3 shows the top 30 geo-influencers (those in bold match the ground truth).

Our approach carries as many as four matches in the top five and as many as twelve matches in the top thirty geo-influencers. This compares favorably to the results reported in [42]: two matches in the top five and eleven matches in the top thirty influencers. None of our ranked lists incorporate high-level influencers such as @YouTube. Furthermore, our approach allows targeted city collection which results in an overall network being orders of magnitude smaller. As was shown in Fig. 3.2

Table 3.3: Top 30 Geo-Influencers extracted from each Dataset

D1	**D2 News**	**D3 Sports**	**D4 News**
marty_walsh	boston25	**cityofboston**	visitbostoncity
cityofboston	**7news**	hiddenboston	**cityofboston**
bostondotcom	**wcvb**	**wcvb**	**bostondotcom**
bostontweet	**bostondotcom**	**marty_walsh**	**marty_walsh**
mbta	**bostonpolice**	eatboston	bostontweet
onlyinbos	onlyinbos	**bostondotcom**	mbta
7news	massstatepolice	bostontweet	bostonmagazine
bostonpolice	bostonglobe	mbta	**bostonfire**
bostonmagazine	mbta	bostonmagazine	**7news**
massgovernor	**cityofboston**	boston25	**bostonpolice**
wcvb	**marty_walsh**	massstatepolice	mayortommenino
hiddenboston	massgovernor	massdot	massdot
boston25	bostonmagazine	**7news**	eatboston
bostonglobe	bostontweet	**bostonfire**	bplboston
massstatepolice	**wbz**	massgov	**wcvb**
bostinno	985thesportshub	theimproper	bostonglobe
bostonpwd	**nhlbruins**	**bostonpolice**	massgovernor
nhlbruins	scottzolak	bosbizjournal	massgov
stoolpresidente	jerry_remy	**wbz**	boston25
edelman11	stoolpresidente	mbta_alerts	massstatepolice
wbz	wilfork75	massema	bostonparksdept
theimproper	hiddenboston	mbtatransitpd	bostinno
bostonfire	justamasshole	eaterboston	theimproper
redsox	lowellsunnews	massgovernor	hiddenboston
celtics	**celtics**	harveywcvb	**wbz**
bos311	edelman11	universalhub	visitma
universalhub	bostinno	mayortommenino	universalhub
bostonbtd	theimproper	bostinno	onlyinbos
jerry_remy	toucherandrich	985thesportshub	bostoncalendar
bostonherald	nesn	fredtoucher	bostonschools

Table 3.4: From each Dataset Top Geo-Influencers containing keyword Celtics

D1	**D2 News**	**D3 Sports**	**D4 News**
celtics: 25	celtics: 25	bdcceltics: 149	celticsblog: 304
celticslife: 525	celticslife: 115	nbcsceltics: 199	nbcsceltics: 386
		celtics: 226	celticslife: 424
		r_bostonceltics: 261	bdcceltics: 483
		celticsviews: 293	celtics: 975
		celticsfanclub: 596	

there are at most 100,000 users collected per city.

A number of geo-influencers overlap across the four datasets. This again reinforces that similar results can be achieved via differing city communities as long as those communities are local to the city. What differentiates communities is that the ranking will be slightly tilted towards the query. For example, Table 3.4 shows geo-influencers, and the position in the ranked list of 1000, that contain the keyword 'Celtics' (a popular basketball team associated with Boston). As expected, D3 produces the most sports related influencers (because query also contained 'sports').

3.6 Geo-Influencer Collection Runtime

Typically it is assumed that the social graph has already been collected and then different algorithms use its structure and various features in an attempt to identify the most important nodes. The algorithms are evaluated based on accuracy and runtime. We noticed that the algorithm runtime is negligible compared to the data collection time i.e. it is not uncommon for a researcher to have spent a year collecting the dataset, a dataset that cannot be fully shared with others (only unique ids can

be shared and crawling on ids can take about as long as starting a new collection from scratch), and the dataset may not work all that well for a scenario that targets a certain demographic.

Up to 1% of all Twitter message traffic can be collected using the unfiltered stream of tweets (from our experiments around 4 million tweets a day). This may seem like a lot of data, but for a specific use-case such as the 2014 annexation of Crimea by Russia, there might not be that many messages from the area of interest. The method proposed in this chapter is useful for identifying influencers and corresponding user communities from a particular geographic area. As an illustration, we show the expected collection times for three methods:

- M1: using the union of followers from multiple geo-influencers
- M2: using followers of a well-known influencer (national or global like)
- M3: using message traffic

For influencers, the self-reported locations of the followers are analyzed. For message traffic, the self-reported locations of the users that generated the message are analyzed. For this illustration, we are interested in forming communities for cities: Syracuse and Buffalo of USA. The users whose self-reported location matches one of the cities are recorded. Each self-report location is turned to lowercase and it is checked whether the city name is present within it.

The geo-influencers associated with a city are found via automated Google search. All of the Twitter influencers identified in the top 100 URLs by Google are utilized.

All of the followers for these geo-influencers were collected for each city; for Buffalo[2], a total of 2767322 were processed vs. 1165345 for Syracuse[3] (some followers repeat across geo-influencers, number of unique followers were 1774172 vs. 566815, respectively). The collection time was 29.3 hours for Buffalo and 13.16 hours for Syracuse. Fig. 3.5 shows the number of unique followers that contain the city name in their self-reported location per 25000 followers for the two communities. The number of new followers with Syracuse in their self-reported location plateaus at around 20K. While the followers that contain Buffalo in their self-reported location continue to rise to 55K.

Using influencer's followers it takes around 110 seconds to process 5000 followers[4]. So theoretically, 3.92 million followers can be collected and processed daily. There are other factors impacting collection such as how quickly one can perform preprocessing and writes to a database which is why our collection times are a little different.

For method M2, using global/national like influencer, @NPR is used with 7.054M followers collected. This influencer was chosen because it is popular in the USA (a more global influencer like @CNN will perform worse as it will capture a smaller

[2]Followers collected over Buffalo geo-influencers (28 total): DAErieCountyNY, NWSBUFFALO, BfloBizFirst, SPECNewsBuffalo, WBFO, BPDAlerts, RedandBlack716, BFLO_CC, Buffalo_Schools, SURJBuffalo, markpoloncarz, USACE_Buffalo, BuffaloSewer, BuffaloBills, MobBuffalo, BuffaloSabres, wnymedia, IIBuff, news4buffalo, BuffaloNiagara, ECDOH, WGRZ, TheBuffaloNews, MayorByronBrown, buffalo_ny, BuffaloEats, FBNY_WNY, WKBW

[3]Followers collected over Syracuse geo-influencers (28 total): VisitSyracuse, SPECNewsCNY, Cusememes, LO_Syracuse, SyracuseAirport, dailyorange, CNYCentral, syrbasketball, SyracusePolice, NYSFair, CuseWBB, chrsbakr, AndrewDonovan, SyracuseOn247, Cuse_Tennis, SyracuseU, Stephen_Bailey1, Cuse_MBB, NewsChannel9, BenWalsh44, OnondagaCounty, Cuse, CuseFootball, AdrienneSmithTV, SyracuseUNews, syracusedotcom, syracuseITC, Syracuse1848

[4]https://developer.twitter.com/en/docs/twitter-api/v1/rate-limits:
GET followers/ids returns 5000 follower ids per minute, GET users/lookup can be used to process 900*100 ids per 15-minute interval or 100 followers per second. 60 seconds to collect 5000 followers ids plus 50 seconds to process ids gives 110 seconds.

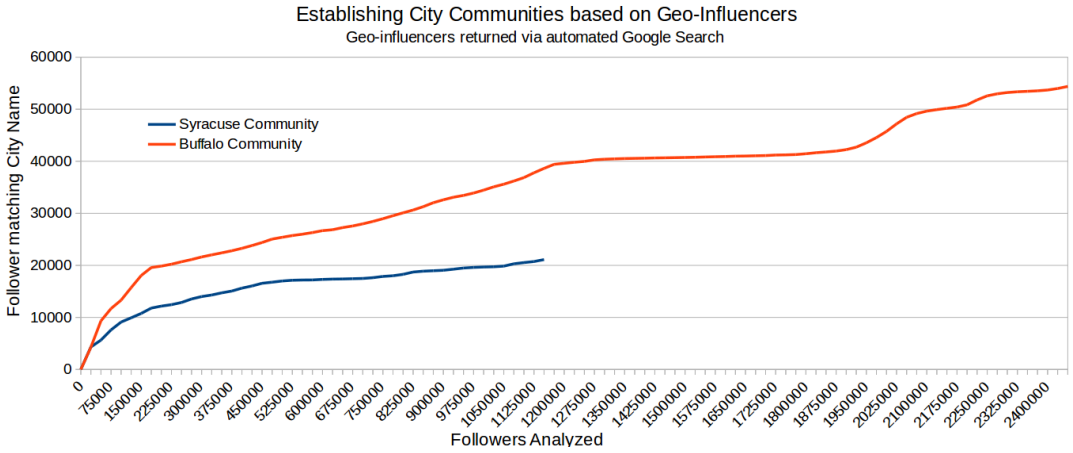


Figure 3.5: There is a large ratio of followers extracted with self-reported location matching the city of interest. Theoretically to process 25K in followers should take approximately 10 minutes. In this way in a relatively short time, in under 24 hours, a community of users that is representative of the city can be extracted. There is higher confidence in these users because they are known to follow a geo-influencer associated with the city as well as having the city in their self-reported location.

Table 3.5: Average Number of Users (per hour) Matching City of Interest

users using method/collection time	**Syracuse**	**Buffalo**
method M1 based on geo-influencers	21240/13.16=1614	56774/29.3=1938
method M2 based on global influencer	1872/64.3=29	3994/64.3=62
method M3 based on message traffic	283/57.36=5	1048/57.36=18

percentage of users for the cities in the USA). For message traffic a total of 10 million messages were collected (around 4 million messages daily). Table 3.5 shows how many users were extracted using each method divided by the total amount of time needed to perform the collection (time in hours). There is a much higher ratio of users for the city of interest from a collection that is focused on geo-influencers using method M1.

Method M1 results in 21240 users for Syracuse and 56774 users for Buffalo.

In comparison using message traffic to generate the same communities would require 21240/5/24=177 days and 56774/18/24=131 days respectively. For Syracuse, method M1 is 1614/5 = 322x and 1614/29 = 56x faster than method M2 and M3. For Buffalo, method M1 is 1938/18 = 107x and 1938/62 = 31x faster than method M2 and M3.

In a separate large collection of users based on followers of verified influencers (tracked by Twitter's @verified) we collected 373 million profiles. This collection took over a year to complete. Across 373 million users, 42448 and 92276 contained Syracuse and Buffalo in their self-reported locations, respectively. This shows that the proposed M1 method can quickly identify roughly half of all Twitter users. The additional benefit in method M1 is that there is higher confidence in these users because they are known to follow a geo-influencer associated with the city as well as having the city in their self-reported location.

Having identified the users that are representative of the demographic area, up to 3200 messages per user can be collected in a relatively quick amount of time[5]. This will provide a larger set of relevant data than using the alternative methods we discussed. For example for Syracuse the 21240 users can be used to collect 15,206,890 messages and for Buffalo the 56774 users can be used to collect 54,913,648 messages. This is much more messages than can be collected in a day using the method M3 and these messages will be relevant to the geographic area of interest. The message creation times can be used to filter to the period of the incident that is wished to be analyzed. The messages can then be used with Natural Language Processing (NLP)

[5]GET statuses/user_timeline allows 900 requests per minute with each request providing up to 3200 Tweets (there is an additional limit to at most 100,000 requests per 24 hours)

to characterize and summarize the situation.

3.7 Conclusions

We have presented a novel method that utilizes geo-influencers for establishing city-level communities and then to identify additional geo-influencers, in a process that can repeat several times. Geo-influencers are at the city-level such as related to the city's mayor, local news, local police, and others. The initial set of geo-influencers is established via automatic Google queries. The followers of these influencers make up the resulting city-level communities; they have an interest in the city and for this reason, continue to follow updates related to the city posted by these geo-influencers. Our method does not require a geocoder to identify users that are local to a city.

We have confirmed that the majority of geo-influencers that Google finds relevant are confirmed via city-level communities built using a geocoder as well as through manual inspection. Communities, made up of users that reside in the city, allowed us to rank thousands of influencers based on how influential they are in a given city. By targeting specific cities our approach can outperform comparable approaches while having an overall network and collection requirements that are orders of magnitude smaller.

Chapter 4

Evaluating City-Level Communities using Location Data

4.1 Introduction

Chapter 3 proposed a method for identifying city-level communities and associated geo-influencers. The method works via: (i) automated Google search queries to get an initial set of city-level geo-influencers, (ii) the followers of these geo-influencers used for forming city-level communities, (iii) the follow connections from communities used for identifying additional geo-influencers. TF-IDF model, where the terms are influencers, generates a ranked list of local influencers for each city-level community. The method was tested using 64 cities of the USA. We have confirmed that there is an overlap between influencers that Google finds relevant and the ones identified via city-level communities built using a geocoder. In this chapter, we perform a more comprehensive evaluation that checks whether the central location from influencer's followers matches the city the influencer is associated with.

From related research, there are numerous approaches for generating a ranked list of location-aware influencers, but evaluation of these is typically performed by human annotators and is thus limited to small datasets. In this research, we fuse information from associations made by Google, links from the Twitter social network, and attributes from user profile information for automatically generating labels. This

allows us to perform an evaluation that covers thousands of location-aware influencers across 763 cities within the USA.

The evaluation can be automated by checking if the city (associated via Google or TF-IDF ranking) of the geo-influencer matches the central location (via location features) from influencer's followers. The features proposed can be used to characterize known influencers in a type of repository that captures the geographic area they serve.

Different ways for assigning a central location are illustrated on (i) Members of Congress for whom the label is the state that the congressman is known to serve and (ii) influencers associated with a city via Google search. The initial set of influencers extracted using Google are evaluated using (i) query (certain queries such as those that contain a city matching a person name are found to be more ambiguous), (ii) number of associated followers (a large enough sample of followers is needed to calculate the central location), and (iii) based on URL position (URLs Google recommends first may have higher confidence).

Important findings of this research are that for cities in the USA: (i) 94.33% of initial influencers returned by Google had their central location match the city being queried, (ii) city-level community that is based on the intersection of followers of multiple city-level geo-influencers is better aligned to the city, and (iii) a classifier for differentiating city-level geo-influencers in the USA vs. national and foreign influencers is possible without a geocoder dedicated to other languages. The approach described in this chapter should be applied to verify that the geo-influencers and the resulting city-level communities for the USA from chapter 3 are accurate.

4.2 Related Research

Location-Aware Influence Maximization (LAIM) aims to rank influencers based on the underlying geographic population [44, 45]. The content posted by these influencers can be analyzed for understanding preferences of the underlying population which can aid in personalized recommendations and targeted advertisement [47]. The followers of influencers can be used for forming communities and understanding how they differ in overall depression [48], crime [49], happiness [50], and other factors.

The current state of the art is to geocode available self-reported locations and infer the rest from friends' locations [52]. Most researches choose to focus on US and English based tweets with user location aggregated at the city, county, and state levels [53]. Language and time zone features are important for differentiating foreign country users [9, 54].

Identifying location-aware influencers typically involves (i) collecting network, (ii) reducing the network to nodes matching the geographic location of interest, and (iii) extracting most important nodes via graph-based measures. Challenge of this approach is that it involves a large collection, sometimes involving billions of messages, that covers a geographical area much larger than is of interest. The method also misses passive users and overexposes itself to actively talking bots [55].

The geographical area of interest is typically specified via a region bounded by some radius R [22, 42]. The users whose home location falls within this radius are then part of the community. Once a community that is representative of the geographical area is established traditional measures such as closeness centrality,

and more recently variations on the PageRank algorithm, are used to extract the most important influencers [21].

The outputs of competing influencer ranking algorithms can be compared via measures such as Spearman's correlation, Kendall's Tau, and Rank Biased Overlap (RBO) [56]. The evaluation of whether an influencer is actually within the geographical area of interest is often neglected. Most often the influencers are assumed to be within the geographical area of interest based on their self-reported location or some other heuristic.

Diffusion model is used for estimating how the influence propagates through the network using Independent Cascade, Linear Threshold, Triggering, or Time Aware models [44]. Simulations helpful for understanding the overlapping effect between followers [57]. Models help evaluate the best set of influencers to trigger a large cascade of further adoptions of a new behavior based on a contagion process [58, 59]. These simulation models may contain mistakes if the influencer is not from the geographical area of interest.

The evaluation of whether an influencer is actually within the geographical area of interest is often limited to manual human efforts. For example, ground truth may consist of influencers that are discovered by human annotators and the algorithm is evaluated based on the percent of ground truth influencers identified [42, 46]. Such evaluations contain human bias and are limited to at most dozens of influencers across a handful of locations.

4.3 Assign Central Location (ACL)

This section describes the approach for automatically assigning a central location to each influencer. Our focus is on high confidence self-reported locations that contain both the city and state abbreviation [11]. Let set C represent 763 US cities as possible values for the central location[1] and $F(u)$ represent set of followers associated with influencer u. $D(F(u), c)$ gives the ratio of self-reported locations from followers of influencer u that map to city c:

$$D(F(u), c) = \frac{\text{followers mapping to city } c}{\sum_{k=1}^{763} \text{followers mapping to city } c_k} \tag{4.1}$$

A lowercase city-state string represents each city c (without whitespace or punctuation), example 'newyorkny'. Each follower's self-reported location is turned to lowercase with punctuation and whitespace stripped out. The preprocessed location is utilized if it matches one of the cities in set C. Examples of self-reported locations that map to city 'newyorkny': 'NewYorkNY', 'New York, NY :)', 'New York,NY'. D(F(u), c) values form a distribution over all cities. As an example top three values for influencer @ChicagoTribune correspond to: 'chicagoil': 0.553, 'washingtondc': 0.026, and 'newyorkny': 0.019. The central city c location for influencer u is given by:

$$C1(u) = c^* \text{ where } D(F(u), c^*) = \max_{c \in C} \{D(F(u), c)\} \tag{4.2}$$

[1]Cities are from US Census Bureau, are representative of all states plus DC, and have a population of over fifty thousand (some exceptions such as Burlington largest city in VT)

$$
\begin{aligned}
C2(u) &= c^* \text{ where } D(F(u), c^*) \\
&= \min_{c \in C} \{ \sum_{k=1}^{763} V(c_k, c) * D(F(u), c_k) \}
\end{aligned} \tag{4.3}
$$

$$
C3(u) = c^* \text{ minimizes } V(c^*, \sum_{k=1}^{763} L(c_k) * D(F(u), c_k)) \tag{4.4}
$$

where $V(c_1, c_2)$ is the Vincentry's distance [19] between coordinates associated with city c_1 and c_k; $L(c_k)$ gives the latitude and longitude associated with city c_k.

$C1(u)$ gives the city c that captures the largest ratio of influencer u's followers. $C2(u)$ gives the city c with the smallest average measure (distance between city and neighbor times frequency associated with neighbor) across all city neighbors. $C3(u)$ is the city c whose coordinates are closest to the average latitude and longitude.

Let $L(u)$ specifies latitude, longitude coordinates at the center of the geographic area that the influencer u is supposed to serve. In future sections, such a label can come from the city that Google associates with an influencer, state associated with a member of Congress, or a city using TF-IDF measure. The error distance (ED) is the Vincenty's distance from the coordinates in label $L(u)$ to computed central location $C(u)$ (where $C(u)$ can be $C1(u)$, $C2(u)$, or $C3(u)$):

$$
ED(u) = distance(L(u), C(u)) \tag{4.5}
$$

Across a set of influencers U we can calculate corpus level metrics (1) Mean ED, (2) Median ED, and (3) ED@X; percent of influencers confirmed by followers within X miles of central location; precisely defined below. Median ED is usually

less sensitive than Mean ED for wildly inaccurate predictions. For ED@X, X = 0 corresponds to exact city matches whereas X = 100 considers city matches within 100 miles[2].

$$MeanED = \frac{1}{|U|} \sum_{u \in U} ED(u) \tag{4.6}$$

$$MedianED = \underset{u \in U}{\text{median}}\{ED(u)\} \tag{4.7}$$

$$ED@X = \frac{|\{u \in U | ED(u) \leq X\}|}{|U|} \tag{4.8}$$

4.4 Verification using Members of Congress

Influencers consisting of 463 members of Congress were used to test the ACL process. At most 500K followers were sampled per influencer. In the dataset so obtained our goal was to check whether the central location computed from Congressman's followers matches the home state that the Congressman is known to serve. For example, if a Congressman is known to represent the state of Nebraska will his followers be concentrated around the same state. The central location for each influencer comes from the ACL process using (i) city distribution and assigning state from the associated city centroid or (ii) directly computing state centroid from state distribution (by aggregating 763 cities into 50 states + DC). Table 4.1 shows performance using ED@0 and Mean ED.

[2]X=100 is a popular distance for categorizing mismatch between the user's self-reported location and location inferred from messages [3]

Table 4.1: Percent of Congressmen whose followers confirm home state. For each central location calculation (ED@0, Mean ED) are shown.

Distribution	**C1**	**C2**	**C3**
city	20.3%, 789.82	**27.65%, 483.02**	4.54%, 596.51
state	**53.35%, 389.38**	33.69%, 452.35	6.05%, 595.74
city (DC off)	**93.3%, 72.43**	61.99%, 227.72	9.72%, 563.32
state (DC off)	**90.06%, 188.43**	66.31%, 227.5	6.91%, 558.23

The first two rows show performance that is penalized due to the majority of members being associated with DC. For example, the C1 column in the first row had 368 out of 463 members (79.5%) associated with DC, but the other 93 out of 95 members (97.9%) were accurately associated with the Congressman's home state. Similarly, C1 column in the second row had 203 out of 463 members (43.3%) associated with DC with the other 246 out of 260 members (94.6%) being accurate. DC is a reasonable point of influence for members of Congress, but we experimented with whether the home state can be retrieved if DC is not an option.

The last two rows show performance if Washington DC is removed from the frequency distribution. The best results were using C1 with city distribution where the home state was matched for 433 out of 463 members. In-depth analysis of 30 Congressmen that did not match their home state showed that either they had a large national following (such as Speaker Ryan, Senator Sanders, and Senator Warren) or were mixed with a neighboring state (for example @senatormenendez, @billpascrell, @frankpallone, @repchrissmith, @replobiondo, @replancenj7, @usreprodney, @reptommacarthur, and @repbonnie represent the state of NJ but most of the followers associated with the state of NY).

This section confirms our hypothesis that location-aware influencers can be identified by the location that captures the biggest percent of the influencer's followers. We observed that C1 performs the best using city distribution. It is important to consider C2 since it compares the city to every other city within the distribution. When C2 matches C1, it means that locations outside of C1 are either clustered around it or do not carry enough weight to shift it. For this dataset, there was a significant discrepancy between C1 and C2 because the Congressman's influence was often divided between DC and Congressman's home state. C3's poor performance highlights that a simple mean of coordinates is not well suited for this problem (C3 was tested on other datasets with similarly poor results and as a result is not mentioned in future sections).

4.5 Features

Geocoded location, time zone, and language are often described as the most important features for differentiating between users [9, 54]. Geocoded location and language were utilized (timezone not used as it became a private field in 2018). In all 20 features were proposed as shown in Table 4.2.

Features have discriminatory characteristics for identifying city vs. global influencers, as evident from values associated with @ChicagoTribune (a city influencer) and @CNN (a global influencer) which will be applied in a classifier in Section 4.9. As described in section 4.3 the influencer's followers' self-reported locations were used for generating a frequency distribution over 763 cities. This distribution was used

Table 4.2: Proposed Features

F	Description	CNN vs ChicagoTribune
F1, F2	Centroid based on most frequent city (C1), Centroid based on mean coordinate (C2),	(losangelesca, stlouismo) vs. (chicagoil, chicagoil)
F3	Vincenty's distance between (C1, C2)	1589.4 mi vs. 0 mi
F4	Number of followers that the influencer has	41275970 vs. 1069351
F5	Number of followers collected	991992 vs. 1060039
F6	Followers from F5 with non-empty location	451897 vs. 414695
F7	Followers from F6 mapping to one of 763 cities	43717 vs. 144580
F8	Percent of followers with non-empty locations used in distribution: F7/F6	9.7% vs. 34.9%
F9, F10	C1 and C2 radius: average Vincenty distance from centroid to all other city sensors within distribution	1440.8, 895.6 mi vs. 314.3, 314.3 mi
F11, F12	Ratio of followers captured by city centroid C1 and C2, respectively.	5.8%, 0.6% vs. 55.3%, 55.3%
F13, F14	Follower sample ratio mapping to centroid: F8*F11, F8*F12 for C1, C2.	0.56%, 0.058% vs. 19.3%, 19.3%
F15, F16	Ratio of followers captured by city/Ratio of population captured by city where city is associated with C1 and C2, respectively	1.8, 2.5 vs. 25.5, 25.5
F17	P-value from Kolmogorov-Smirnov test	0.562 vs. 0.789
F18	ratio captured by English language	81.6% vs. 90.7%
F19	most frequent non-English language	Spanish vs. Spanish
F20	ratio captured by F19	4.5% vs. 3.5%

for calculating C1 and C2 which serve as F1 and F2 features.

City distribution, user profile information, and centroids used for calculating features F3-F14 as shown in the table. Distribution of the population was obtained by dividing the population of a city by the population across all 763 cities (using US Census populations). 763 city distribution and population distribution used for calculating features F15-F17. Finally, the preferred follower's profile language used to generate a language distribution over all influencer's followers for features F18-F20.

4.6 Approach

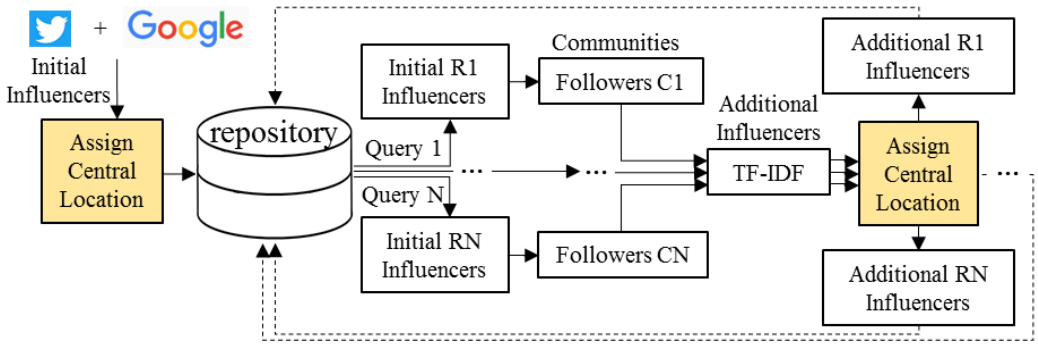


Figure 4.1: Establishing and continuously updating a repository of influencers.

Our proposed solution for building and continuously updating a repository of geo-influencers is illustrated in Fig. 4.1. The process extends the approach from previous chapter via the Repository and ACL process (highlighted in yellow). The required inputs are the geographic regions of interest over which influencers will be collected. A city-level location is the smallest geographic area considered since this is a popular choice among Twitter users [51].

Larger geographical areas can be formed from cities along recognized political boundaries; cities make up fifty states, states combined into nine divisions, and divisions combined into four regions of US. Examples of system input would then be Syracuse NY vs. Buffalo NY (city), NY vs. PA (state), Mountain vs. Pacific (divisions), and Northeast vs. Midwest (regions). This section describes experiments with city and state as regions. R1 to RN geographic regions with N>1 are expected so as to be able to apply the TF-IDF measure. The main processes from Fig. 4.1 are described below:

- Initial Influencers: Twitter influencers from automatic Google searches related to cities making up the geographic regions.
- Assign Central Location (ACL): the central location assigned to each influencer based on the distribution exhibited by the influencer's followers, see section 4.3.
- Repository: stores centroids from ACL and additional features such as percent of followers captured by the centroid, average radius, and others, see section 4.5. Repository also stores whether influencer is local or global to country of interest using classifier from Section 4.9.
- Query: repository queried for local influencers whose C1 centroid matches the geographic region of interest. Each query can contain additional inputs such as the minimum percent of followers associated with the region.
- Communities and Additional Influencers: Followers of influencers from regions R1 to RN form communities C1 to CN, respectively. Influencers that these communities follow are ranked via TF-IDF. Top influencers from TF-IDF go through the ACL process and stored in the repository for future reference.

Additional influencers refine communities associated with each geographic region and the whole process shown in Fig. 4.1 can repeat.

4.7 Analyzing Geo-Influencers Recommended by Google

Initial influencers stem from automatic Google searches. As an example, Table 4.3 shows screennames extracted from top five URLs associated with query 'Syracuse,

Table 4.3: Top Ranked URLs via Google Search for 'Syracuse, NY Twitter'

Hit	URL	Influencer
0	https://twitter.com/syracuse1848?lang=en	syracuse1848
1	https://twitter.com/syracuseu?lang=en	syracuseu
2	https://twitter.com/hashtag/syracuse?lang=en	
3	https://twitter.com/syracuseunews?lang=en	syracuseunews
4	https://twitter.com/syracusedotcom	syracusedotcom

NY Twitter'. The order of returned URLs is recorded. It was expected that the influencers extracted from first web hits will have a higher correlation to the city queried. Fig. 4.2 shows the number of influencers extracted per URL using the top 100 URLs. Queries performed across 763 city state pairs where the number of influencers per city ranged from 1 to 33. Over these cities, there were a total of 14092 influencers, 13050 remained after removing influencers associated with multiple city queries or whose followers had no location information.

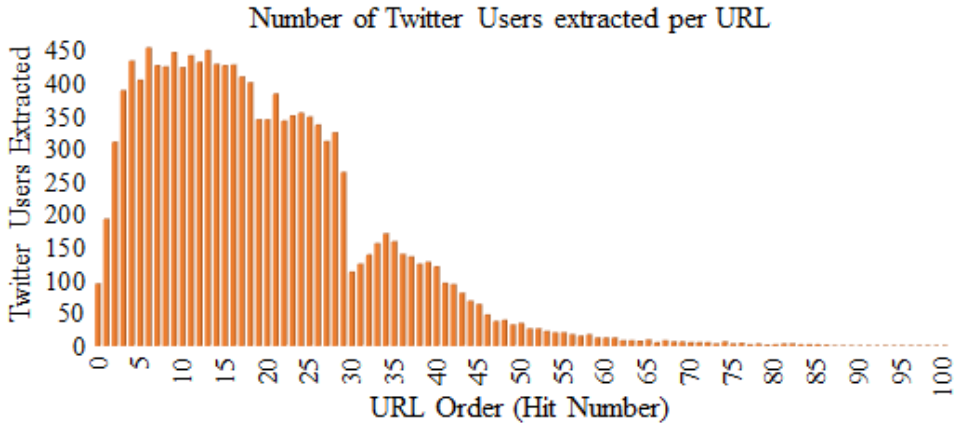


Figure 4.2: Number of Twitter Users extracted via Google using top 100 URLs

Next, a maximum of one million followers was collected for each influencer. Influencer's followers' self-reported locations were used to generate a distribution and

Table 4.4: Average Performance across Google's Queries

Error Measure	**C1**	**C2**
Mean ED	53.89 +/- 164.63	63.17 +/- 150.64
Median ED	19.74 +/- 160.63	16.81 +/- 133.23
ED@0	0.69 +/- 0.32	0.66 +/- 0.29
ED@100	0.95 +/- 0.12	0.92 +/- 0.13

compute centroids as described in Section 4.3. Error distance (4.5) computed between coordinates associated with the city queried vs. the central location coordinates from influencer's followers. Across all influencers ED@0 was 73.58% for C1 and 70.11% for C2. The subsections below examine the performance based on the query type, the follower sample size, and URL order.

4.7.1 Performance based on Query Type

Mean ED, Median ED, ED@0, and ED@100 were calculated across the influencers associated with each city query. Table 4.4 shows the average and standard deviation for C1 and C2 centroids across queries. It is time consuming to analyze thousands of influencers via manual validation, but with the error measures proposed, we can quickly focus in on those queries that are problematic for Google.

Out of 763 queries, only fourteen queries had ED@100 under 50%. Thus most of Google's city-influencer associations are confirmed by the central location from influencer's followers. The problematic queries are described below.

One type of mistake stems from matching influencer not on location but based on

screenname, description, or user specified name. Example of cities with human names are Lawrence MA (out of 8 influencers, city from query matched 0% of self-reported locations, but matched 100% of names example Jennifer Lawrence) and Anderson IN (out of 27 influencers, city from query matched 7% of self-reported locations, but matched 96% of names example Anderson Cooper). Contrast this to Trenton NJ where out of 27 influencers, the city from query matched 78% of self-reported locations but matched only 52% of names.

Another mistake is related to state abbreviations that can be interpreted as a preposition ('OR' and 'IN'). Nine out of fourteen queries faced this challenge: Gary IN, Albany OR, Anderson IN, Lafayette IN, Salem OR, Springfield OR, Gresham OR, Hillsboro OR, and Medford OR. Spelling out the state name might help these queries, i.e. Oregon instead of OR. The state name should not be spelled out for all queries because Google has more data for more common queries. For example, in the previous chapter we saw that typing out 'New York, New York' causes Google's search to favor results associated with a casino in Nevada of the same name. This is again observed in that for 'New York, New York Twitter' @NYNYVegas was the second URL recommended, i.e., spelling out the state name might also bring unintended results.

Finally, there were city names that are not unique to a single state. For example, there are 34 states with Springfield cities. Table 4.5 shows specific instances where the wrong influencer was matched to query Albany OR and associated true location.

Table 4.5: Google Mistakes for Query 'Albany, OR Twitter'

Albany OR Associated Influencers	**True Loc**
albanyairport, albanysym, dutchmenpgcbl, reinventalbany	Albany, NY
naschoolupdates, newalbanyohio	Albany, OH
ahshuskies, ahuskiebaseball	Albany, MN
albanyassociate, albanymusicuk, thealbanyse8	UK

4.7.2 Performance based on the follower sample size

As described in Section 4.3 the central location is computed from influencer's followers' self-reported location distribution. For a small number of followers, there might not be enough locations to generate a proper distribution and associated centroid.

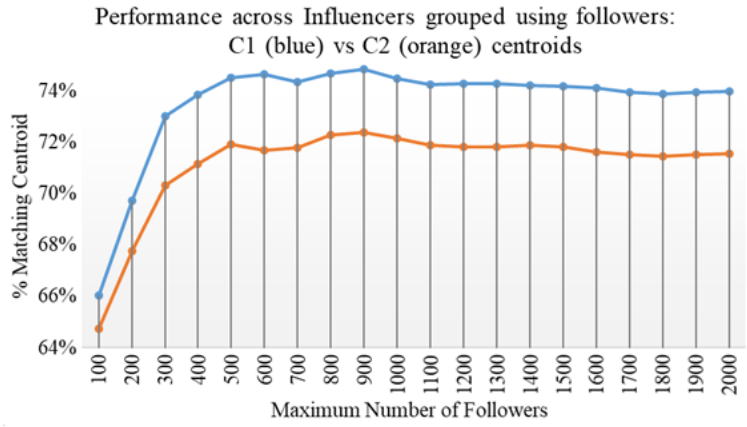
Fig. 4.3 (top) shows ED@0 error for C1 and C2 centroid as influencers with an increasing number of followers are considered. The figure illustrates that at least 500 followers are needed to get a large enough sample for computing the centroid. Because most of the influencers are associated with city level locations they, in general, cater to a smaller audience: 20% had 500 and 54% had 2000 followers or less.

4.7.3 Performance based on Query Result Order

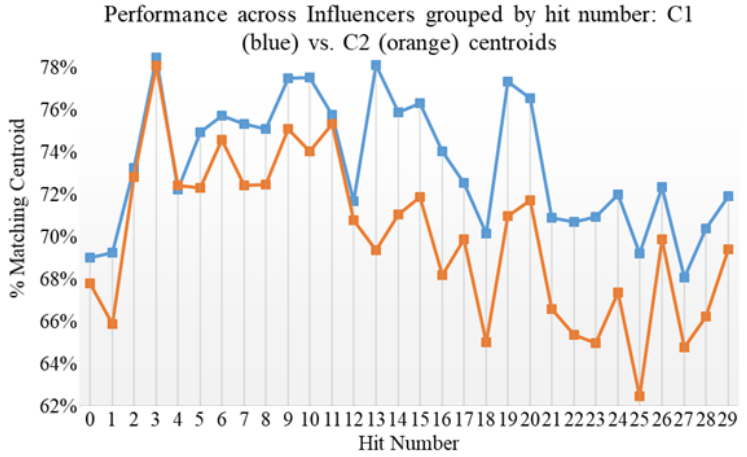
First web hits have a much higher click-through rate. As a result, it was tested whether the influencers that are in the top results (low hit number) would have a higher accuracy in being associated with city query.

Fig. 4.3 (bottom) shows ED@0 error for C1 and C2 on influencers grouped by hit number 0 to 29. Top 30 web hits were chosen because each had a good sample of

Twitter influencers ranging from 387 to 490. The figure shows that the results remain about the same for influencers that appear in top vs. later web results. A possible explanation is that for ambiguous queries Google will have poor results across the board.



(a) Influencers grouped by number of followers



(b) Influencers grouped by URL position from Google search

Figure 4.3: **Top** – Performance based on Follower sample size. 500 or more followers provide a good sample for centroid calculation. **Bottom** – Results from top 30 URLs illustrate that influencers in top web search results have similar performance as influencers in later web results.

4.7.4 Forming Optimal Communities

The higher the concentration of followers associated with a geographic city the better they are for establishing a city community. As an example, Table 4.6 (top) shows five influencers recommended by Google for Syracuse, NY (each confirmed by their respective C1 centroid). Out of these @syracusedotcom has the highest concentration of followers associated with the city (70.1%) and is thus the best for establishing the associated city community. Table 4.6 (bottom) shows the percent of followers from Syracuse NY that follow a pair of influencers. Despite @cuse_mbb having only 21.5% of its followers from Syracuse, the table shows it can be used to improve percentages associated with followers extracted from other influencers. In this way, if a researcher wanted to focus on users from Syracuse that are interested in basketball and news, then followers of @cuse_mbb and @syracusedotcom could be chosen to establish the city community with 73% of followers mapping to Syracuse.

A user that follows two geo-influencers has a higher chance of being from the city than the one that follows a single geo-influencer. We analyzed the percent gain over all possible pairs of influencers, where both influencers were associated by Google with the same city and confirmed by C1 centroid from the ACL process. There were 15951 pairs that produced 500 or more mutual followers across 492 cities. On average the pair had an 11.1% gain over a single influencer. 3275 pairs had $90-98\%$ percent of followers matching city of interest across 137 cities. Focusing on these pairs would lead to better city communities. It is not recommended to use three or more influencers because the overlap in followers may be too small.

Table 4.6: Percent Followers mapping to City for Single vs. Pair of Geo-Influencers

Percent Followers of a single Geo-Influencers mapping to City

Influencer	Number Followers	%C1
syracusedotcom	87334	70.13
sucampus	10238	53.23
gosyracuseu	2495	45.65
cuse	138595	41.63
cuse_mbb	261805	21.49

Mutual Followers of Two Geo-Influencers better aligned to city

Influencer Pair	Mutual Followers	%C1
syracusedotcom + @cuse_mbb	2784	73
sucampus + @cuse_mbb	415	55
gosyracuseu + @cuse_mbb	396	67.5
cuse + @cuse_mbb	3250	48.7

4.8 Classifier for City-Level Geo-Influencers

In this section, we generate a classifier for differentiating US city-level geo-influencers vs. influencers that are from foreign countries or have more global influence. Our approach illustrates that it is possible to differentiate the two types by only geocoding the locations associated with the USA.

Our dataset contained a total of 8740 influencers: 350 global influencers vs. 8390 US city geo-influencers. 8390 geo-influencers are obtained from Google and TF-IDF ranking. The city that the geo-influencer is associated with is verified by C1 centroid from the ACL process, the associated city name is also within the influencer's self-reported location, and each influencer had at least 500 followers (to ensure a large enough sample size as discussed in Section 4.7.2). 350 global influencers obtained via manual Twitter searches: 250 influencers are popular worldwide such as

webmd, spacex, twittersports; 100 influencers came from foreign countries such as (screenname: country): ttcnotices: Canada, dailysabah: Turkey, vesti_news: Russia, live_hindustan: India, greateranglia: UK, and others.

Numeric features F3-F18 from Table 4.2 were normalized to between 0 and 1 range. Nearest Neighbor (k=20), Gaussian Naïve Bayes, Decision Tree, Random Forest, and Support Vector Machines (SVM) with a linear and radial kernel were tried as classifiers with 3 fold cross validation[3]. Unbalanced classes were handled by providing weights to SVM and random forest classifiers; global influencers were weighted 0.999 vs. 0.001 for city influencers (i.e. an incorrectly classified global influencer penalized classifier 1000 to 1 to ensure that all global influencers are accurately classified). We also tried to balance out the dataset by random over-sampling of the minority class. SVM and Nearest Neighbor were the only classifiers which classified all global influencers accurately. Fig. 4.4 shows average accuracy for these classifiers using an increasing number of features (drop in accuracy is due to the USA geo-influencers being classified as global/foreign) . Features ranked through Recursive Feature Elimination (RFE) with linear SVM.

All three classifiers have peak performance when using the top four ranked features: F8 (percent followers mapping to US distribution), F13 (ratio of followers from sample mapping to C1 centroid), F7 (number of followers to US distribution), and F3 (distance between C1 and C2). Classifier performance using these four features illustrates that it is possible to differentiate US city vs. global influencers without having to geocode locations outside of the USA.

[3]Scikit-Learn package utilized for implementation: https://scikit-learn.org/

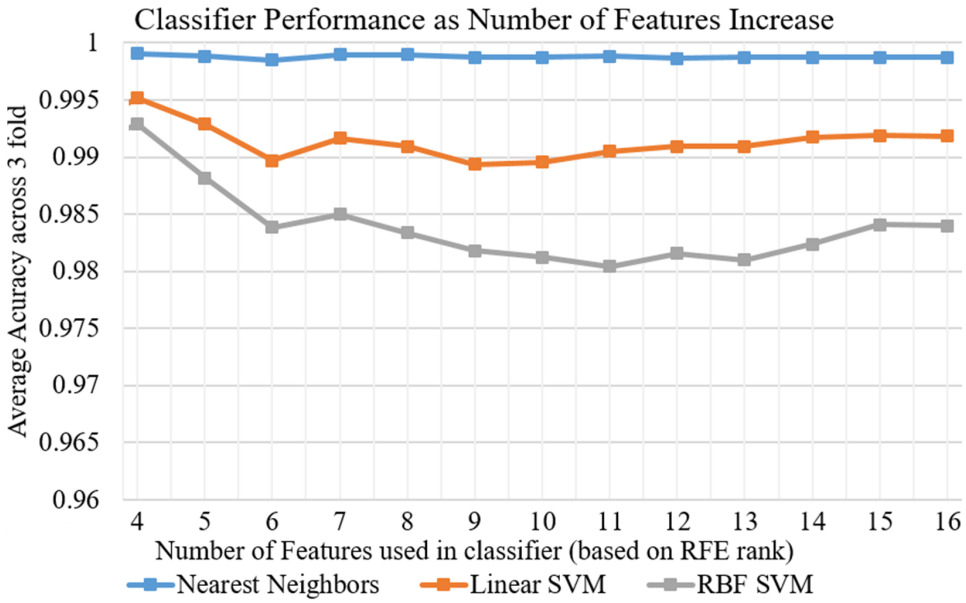


Figure 4.4: USA vs. Foreign Country classifier. Overall performance across three classifier peaks using top four ranked features.

The results are intuitive in that influencers associated with a city c in a country x should (i) have a higher overall concentration of followers going to this country x (necessary for filtering out influencers associated with foreign countries) and (ii) should exhibit an above average concentration of followers associated with the specific city c (necessary for filtering out global influencers that may have a high concentration of followers in country x but whose influence spreads over many cities).

These are important results to consider because geocoding locations all around the world is difficult. For a specific country of interest, our recommendation is to focus on well-known cities within this country for forming a frequency distribution as was described in section 4.3. City influencers can then be extracted using a classifier and by focusing on those influencers verified by the C1 centroid. As the repository continues to grow in size the classifiers are expected to continue improving.

4.9 Conclusions

Our research proposed an automated evaluation for a targeted collection of influencers and corresponding city-level communities. The approach, described in this chapter, should be applied to verify that the geo-influencers and the resulting city-level communities for the USA from chapter 3 are accurate.

The evaluation showed that Google does occasionally make mistakes for queries involving ambiguous city names (those that appear along with multiple states or that match popular human concepts). Our evaluation process allowed us to quickly identify these errors without having to review thousands of influencers manually. Queries with fewer than 50% of influencers within 100 miles of the expected centroid (ED@100) were manually verified to be challenging for Google. The performance was about the same for influencers in top vs. later web results, i.e. web hit number does not play a significant role in how well influencer is associated with city query. Finally, it was shown that at least 500 followers are needed to have a large enough sample from which to compute the central location.

The method allowed to automate an evaluation covering thousands of influencers. Larger geographical areas were specified by aggregating multiple cities for a state-level evaluation. It was also illustrated how multiple influencers with a geographically local audience could be used to form city communities better aligned to the location of interest. Finally, a classifier was proposed for differentiating the USA vs. global and non-USA influencers; this classifier is possible without a geocoder dedicated to other languages.

The methods described here are useful for generating and maintaining a repository of city-level influencers for the USA or other English-speaking countries (this is because the evaluation is still reliant on a gecoder that can process English-based locations). In the next chapter, we describe an approach that works worldwide by categorizing influencers using time-based features.

Chapter 5

Inferring Degree of Localization and Popularity of Twitter Topics and Persons using Temporal Features

5.1 Introduction

Previous chapters have focused on improving geocoding and leveraging Google search for associating influencer with a city. In this chapter, we illustrate an alternative approach for how the creation times can be used to infer geo-information.

On Twitter, every user and every message has a creation timestamp. For a group of users or a group of messages, the creation times can be used to help determine whether the group is concentrated in a single time zone or is spread out more globally. The Coordinated Universal Time (UTC) offset[1] can be identified for a group that is from a specific time zone. For a global group (such as the followers of a global influencer), the daily changes in followers can be inferred and used for studying the influencer's evolving popularity.

The time-based features discussed have applications related to (i) local expert finding in social networks, (ii) inferring when followers joined an influencer, and (iii)

[1]UTC is the time standard used globally, defined by the International Telecommunication Union Recommendation (ITU-R TF.460-6); it is a refinement of previous time standards such as Greenwich Mean Time. For instance, the UTC offset is -5 for the time zone that includes the northeastern USA.

understanding popular trending topics from message traffic relevant to a specific geographic area. The methods in this paper maintain user's privacy because the location inference is at the timezone level.

When performing Twitter data collection need to consider Twitterbots, a software program that sends out automated posts on Twitter [43]. There are malicious and benign bots. Malicious bots threaten the security of other users [60] by posting malicious URLs along with hot trending topics [61]. Such accounts are actively being blocked by Twitter. Examples of benign bots are job postings, weather, news, and traffic updates. Such bots do not violate the rules of Twitter and are allowed to operate. The issue is that the bots can generate a lot of message traffic compared to real users. For example, Tasse et al. [15] find that job-posting bots constitute a growing portion of the public geotags. For analysis over message traffic, to reduce the impact of bots, it is recommended to focus on a single, most recent, message per user. For analysis over influencer's followers, it is recommended to consider each follower's tweet frequency (number of messages posted by follower divided by the number of days elapsed since account creation). Our analysis is focused on influencers that have been verified by Twitter to be legitimate but in general followers-friends ratio, tweet frequency, number of times added to favorites, and other features such as screen name length are used for identifying real influencers [62].

The rest of the chapter is structured as follows. Section 5.2 reviews prior research related to local expert finding. Section 5.3 shows how group creation times can be used in a time distribution and how this distribution can be used for predicting the UTC offset. Section 5.4 analyzes the temporal distribution of message traffic. Section

5.5 analyzes the variations in the number of followers and illustrates how those can be used for understanding daily followers gained. This is useful for link inference and understanding evolving popularity of global influencers. Section 5.6 describes a classifier for the discrimination of local vs. global influencers. Finally, Section 5.7 presents our conclusions and future research directions.

5.2 Related Research

The problem of finding authoritative users is known as expert finding; this is a well-studied problem with research going back over a decade, and has gained popularity within the information retrieval community since it was included in the TREC enterprise track [63]. A recent survey by Husain et al. [64] reports that a majority of the expert finding systems were used in: (i) the academic domain (research collaborations), (ii) enterprise (experts for offering formal help related to development), (iii) medicine (medical experts), (iv) online knowledge sharing communities, (v) online forums, and (vi) social media (finding experts from various social networks like Twitter and Facebook).

Expert finding methods assume that individuals' published documents are relevant to their expertise with different degrees of a match, and they focus on modeling the associations between these documents and candidate experts.

Lappas et al. [65] give an early survey on expert finding in social networks, which typically involves (i) using text content posted by expert candidates and (ii) using the expert candidates' online social connections. Two best-known algorithms that

exploit link structure to find authorities are based on PageRank [66] and Hyperlink-Induced Topic Search (HITS) [67].

Weng et al. [40] proposed TwitterRank which employs the Latent Dirichlet Allocation (LDA) model to detect the topics of individuals based on their tweets. Then, for each topic, it builds a weighted graph based on the topical similarity between two users and then employs a PageRank algorithm to find topic-specific influential users.

Romero et al. [68] designed an algorithm similar to HITS named Influence Passivity algorithm to quantify the influence of users in a Twitter network. This algorithm utilizes both the structural properties of the network as well as the diffusion behavior among users. Pal et al. [69] proposed an attribute-based approach for identifying experts and potential experts in community question answering. Fifteen features were extracted from the Twitter graph and tweets posted by the users, to estimate their levels of expertise on various topics. Clustering (based on the Gaussian mixture model) was used to determine experts, maximizing the likelihood of the data given a number of Gaussian components.

Ghosh et al. [70] proposed a system called Cognos, which represents each user by the metadata of Twitter lists that contain the user, then ranks users based on the similarity score between each user and a topical query. Cognos tends to choose users that are contained in many lists and whose metadata contains the query. The authors show that their system can identify top users for a particular topic better than graph based approaches.

Separately, research efforts have addressed the task of finding local experts with specialized knowledge focused around a particular location. Local experts are important for many applications such as answering local information needs [71].

Li et al. [73] proposed applying points of interest (POI) as a possible categorization of expertise related to a particular geographic location. Example 'Chinese Restaurants' in Los Angeles is a POI topic. High-ranking candidates should be able to answer questions about the locations or the category of locations in the topic. The time user reported being at a POI is seen as an important feature in that frequent visits result in greater familiarity with the location in question [74].

Niu et al. [75] introduced a learning-based method to find local experts on Twitter. They defined multiple classes of features that could impact a user's local expertise, such as tweet content features (e.g. the TF-IDF score of a topic keyword in the candidate's tweets) and local authority features (e.g. the distance between the candidate and the query location). Authors found it best to retain only the first check-in during a repeated activity (a user posting multiple times about a newly served dish during the same meal is an example of the same venue during which the user remains in an unchanged location and activity).

A recent review by Yochum et al. [76] analyzes systems that recommend items (such as venues, places, travel routes, activities, friends, or social media) to users while considering geographical preferences. They analyzed 178 journal papers in this area from 2001 to 2018. They found that Foursquare, Gowalla, Brightkite are popular social media sites since these are Location-based Social Networks (LBSNs). LBSN websites are where users share their locations by checking-in so there is no need to

geocode, geoparse or geotag. Twitter used in about 4.5% of publications vs. 46.6% over these three LBSN sites. Twitter is typically used for getting the popularity of points of interest or locations by extracting from messages with coordinates: (i) construct an ordered sequence of relevant text; (ii) map to the popular points of interest using latitude and longitude; and (iii) generate time sequences of point of interest visits.

Several research papers rely on geotagged tweets or text-based Location Indicative Words (LIW). Singh et. al. [79] focused on tweets with GPS coordinates that contained the words 'flood', 'water', and 'Baarh' for flood event detection. Luceri et. al. [77] propose a deep learning architecture that aims to infer the geo-tag of a generic user's tweet by leveraging the geo-tags shared by other users on Twitter. This work is similar to inferring a user's location based on friends' self-reported locations [13], but instead of using self-reported locations, it focuses on those friends that have generated a message with precise coordinates. To preserve privacy, the authors recommend either to stop producing messages with geo-tags or to purposely alter the geoinformation so that it is outside of the user's actual location. Paule et. al. [80] perform geotagging of tweets using weighted majority voting of geotagged tweets whose content is most similar. This increases available geotagged tweets with improved performance demonstrated in New York and Chicago. In papers that attempt to identify topical experts typically the GPS coordinates and place mentions associated with messages are utilized. Inkpen et al. [81] develop a city, province, and country classifier for monitoring places mentioned in Twitter messages.

The issue with focusing only on tweets with GPS coordinates or POI information

is that they make up a small portion of the Twitter API stream [3, 10]. Geocoding the message's author self-reported location is complicated. Jurgens et al. [10] reported that using popular gazetteer solutions GeoNames, DBPedia, GeoLite, and Google's geocoder were able to each geocode under 4% of users using self-reported location [10].

Multiple surveys have been written related to Twitter user geolocation [3, 10]. Jurgens et al. [10] reimplemented some of the state-of-the-art models, tested and trained them using their own constructed dataset to ensure fairness of comparison, and found significant performance issues. Mourad et al. [72] proposed a guide for a standardized evaluation of Twitter user geolocation. Analysis of fifteen models and two baselines illustrated that the choice of effectiveness metric can lead to diverging conclusions. Due to the high levels of noise and the data collection restrictions imposed by the Twitter API the user geolocation remains an unsolved research area.

Other features useful for identifying locations are the time zone and UTC offset [86, 87]. Zannettou, et al. [88] used time zone information to understand the audience targeted by tweets from Russian-linked accounts. But due to privacy reasons, Twitter has made these fields inaccessible in 2018.

Twitter does not keep track of any time information other than identifying when a user or a message was created. Data for link creation times between users and their followers are not stored, although it can be extracted by performing multiple scans of the Twitter network. For example, Kwak et al. [85] collected daily snapshots of the online relationships of 1.2 million Korean-speaking users for 51 days as well as all of their tweets to estimate popularity dynamics.

This research proposes new time-based features based on user and message creation times. Creation times over influencer's followers are used for predicting the time zone's UTC offset and associated geographic area that the followers belong to. When applied over message traffic, the approach can differentiate top trending topics and persons in different geographical regions. The degree of localization ("localness") is an important concept, with ongoing work in formalizing the notion [54]. Our time-based features are successfully applied in a classifier for predicting local vs. global influencers. The resulting classifier can be applied as a post-processing step for verifying that the local expert is indeed local. The new time-based features are not just limited to inferring location, but can also be used for inferring link creation times for studying the evolution of influencer's popularity.

5.3 UTC Offset Prediction based on Account Creation

This section describes how the time zone's UTC offset is predicted from a set of creation times. The creation times can come from a set of users or a set of messages. Subsection 5.3.1 describes the dataset; the creation times come from a group of users whose self-reported location is in common and where the location's UTC is known. Subsection 5.3.2 describes how a time distribution is formed and how it is used to predict the UTC offset. Subsection 5.3.3 describes experiments to find the optimal parameter values used in the proposed approach.

5.3.1 UTC Offset Dataset

Over 373 million user profiles were analyzed and user groups were chosen based on self-reported location in common. All self-reported locations were turned to lowercase with punctuation and spacing stripped out. Of particular interest are those self-reported locations that match (i) (City, Province) or (ii) (City, Country Name) in English from GeoNames. The city, country pairs are checked to be unique in that there are no other cities within the country with the same city name. The population of all cities considered in is over five thousand. Major well-known city names are included (without the country name) provided the city is unique and has a population of over 1 million. Each self-reported location had to be used by at least 250 unique users to ensure a large enough sample size.

The resulting dataset, denoted D_{UTC}, consists of 12,271 groups. Table 5.1 shows the five most popular locations, the number of users making up each group that use the location, and the UTC offset associated with the location, denoted as UTC^L, using equation (5.1).

$$UTC^L(loc) = \frac{1}{3}UTC(tmz(loc)) + \frac{2}{3}DST(tmz(loc)) \tag{5.1}$$

GeoNames is used to get the location's time zone[2] via function *tmz*. UTC and DST functions are used to obtain the UTC offset during standard and daylight saving time, respectively; these are equal in time zones where daylight saving is not observed. Daylight saving time is typically observed for eight months of the year and is thus given a larger weight.

[2]download.geonames.org/export/dump/timeZones.txt

In our dataset, UTC^L takes 42 possible values ranging from -9.9 to 13.53. Therefore, the corresponding UTC offset interval for our dataset is [-10, 14) (UTC offset -12 and -11 exist, but belong to sparsely populated islands and therefore not of interest). Table 5.1 describes the attributes of the five largest user groups in the dataset.

Table 5.1: Five biggest user groups in UTC Offset Dataset

Location	**Group Size**	**UTC^L**	**Country**
london	2065562	0.667	GBR
losangelesca	1768898	-7.333	USA
newyorkny	1425330	-4.333	USA
chicagoil	1173340	-5.333	USA
parisfrance	1026459	1.667	FRA

5.3.2 Sleep Cycle and UTC offset Determination

The following procedure is used to identify the UTC offset in the geographic area from which the creation times originate. Given a set of creation times:

1. Creation times to Time Distribution:
   (a) The hour from each creation time is used to generate a histogram, with 24 bins corresponding to 24 hours.
   (b) Time distribution refers to a normalized histogram; $f(t)$ used to denote the relative frequency of creation times within t^{th} hour.
2. Preprocessing:
   (a) The 24-hour time distribution is duplicated to generate a 48-hour distribution.

(b) The distribution is smoothed by computing the moving average of $n = 5$ consecutive points.

3. Sleep cycle identification:

   (a) If there are four intersection points (between $f(t)$ and the $p = 33$ percentile), per 48 hours, the sleep cycle is identified as a single continuous segment between two consecutive intersection points where the first has a negative and the second a positive slope.

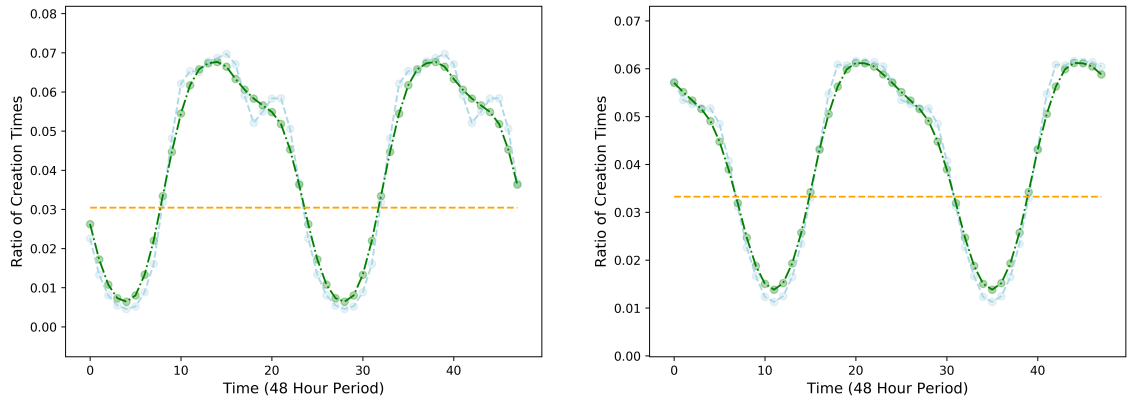
   (b) A quadratic function is fitted over sleep cycle: $f(t) = c_0 + c_1 \times t + c_2 \times t^2$. If $c_2 > 0$, its minimum is considered to be the group's *Potential Sleep Time (PST)*, subtracting 24 if needed, so that PST $\in [0, 24)$.

4. UTC offset computation:

   (a) Given a PST ≥ 14 the transformation PST-24 is applied to transform PST from [0, 24) range to the UTC range [-10, +14).

   (b) Linear regression on known data is used to express the UTC offset as a linear function of PST, using Equation (5.2) at the end of this section based on Fig. 5.3.

As an illustration, Fig. 5.1 shows the $f(t)$ formed from creation times corresponding to users associated with locations (a) 'london' and (b) 'losangelesca'. The data (blue lines) is noisy, and to achieve smoothness we compute moving averages (with $n = 5$ consecutive points), depicted by green lines.The orange line corresponds



(a) self-reported location 'london'

(b) self-reported location 'losangelesca'

Figure 5.1: Normalized 48-hour histograms using creation times of users from (a) 'london' and (b) 'losangelesca' are shown. The blue curve shows the original time distribution, and the green curve represents the moving average (with $n = 5$). The orange line corresponds to the threshold below which the potential sleep cycle is identified from the green curve. The mins between the two charts are 7-9 hours apart matching expectation in that the time difference between the two locations is 8 hours.

to the threshold below which the potential sleep cycle is identified from the green curve.

It is assumed that the regions around the minima (in the smoothed curve) correspond to a nocturnal period when many residents of the region sleep, and hence are not active on social media. This region, expected to be an 8-hour period (a third of the 24-hour cycle) is identified using the threshold $p = 33\%$ in Fig. 5.1). The portion of the smoothed curve below the threshold can be approximated by a quadratic function. Minimum of the quadratic used to predict the UTC offset; confidence in which increases with the coefficient of determination R^2 and the magnitude of the power coefficient c_2 (c_2 close to zero associated with a flat like sleep cycle with not as clear a minimum). We record (i) the predicted UTC offset, (ii) the power coefficient

c_2, and (iii) the coefficient of determination R^2.

The next subsection addresses the selection of parameters for the moving average n and the percentile p threshold, and describes the linear regression leading to the computation of UTC.

5.3.3 Parameter Determination

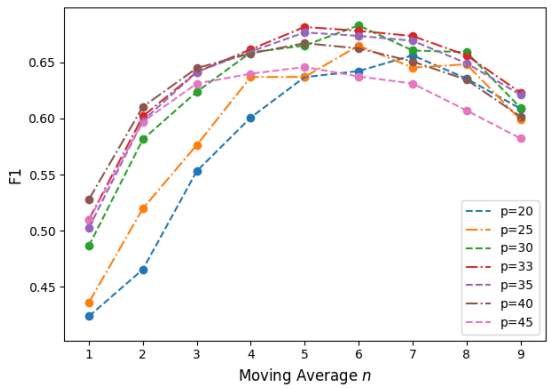
Instead of using the entire $|G|$ creation times of the group, we use a method akin to bootstrapping [82]). Random samples of size M are drawn from G, N times, and for each sample, the PST is calculated. Over N trials, the average PST is denoted $\mu_G(PST)$, and $\sigma_G(PST)$ denotes the standard deviation.

These estimates depend on the choices of the sample size M, the number of samples N, the size of moving average window n, and the sleep cycle threshold percentile p. We performed multiple experiments, with values of $M = [100, 250, 500, 1000]$, $N = 100$, $n = [1, 2, ..., 7, 8]$ and $p = [20, 25, 30, 33, 35, 40, 45]$. Linear regression was performed for PST vs. UTC^L using least squares estimation, as shown in Fig. 5.3. To measure the performance of selected values of the parameters *Recall, Precision,* and *F1* measures were calculated:

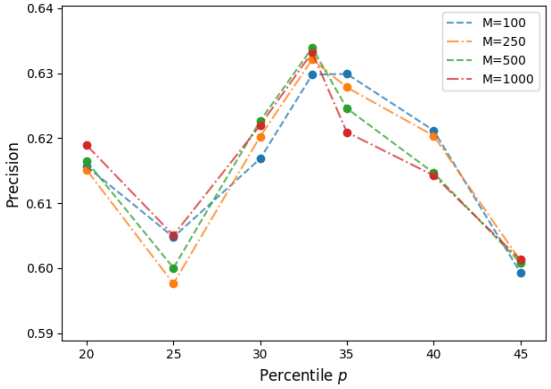
$$Recall = \frac{\text{\# of user groups where PST-estimate calculated}}{\text{the number of user groups}},$$

$$Precision = \frac{\text{\# of correct UTC predictions}}{\text{\# of UTC predictions}},$$

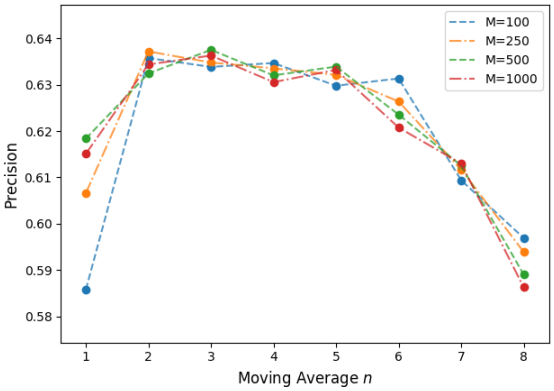
$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}.$$



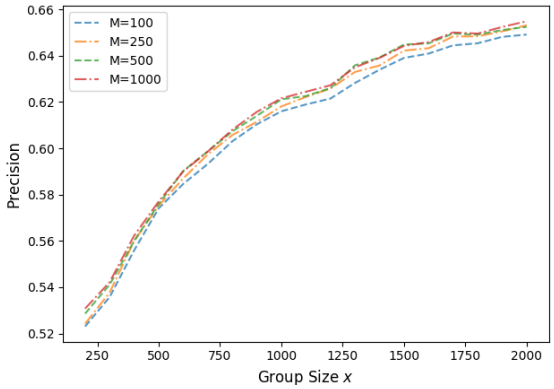
(a) F1 for n and p



(b) Precision for p and M



(c) Precision for n and M



(d) Precision for group size x and M

Figure 5.2: Variation of ability to predict UTC^L ($t_1 = 0.5$) with parameter values: (a) F1 vs. moving average window width n, for different values of percentile p, fixing $M = 250$; (b) Precision vs. percentile p, for different sample sizes M, fixing $n = 5$ which yielded the best F1 score; (c) Precision vs. n for different values of M, fixing $p = 33$ which yielded the best F1 score; and (d) Precision vs. group size x for different values of M, using sampling with replacement, and fixing $p = 33$ and $n = 5$.

Predictions that were more than $t_1 = 0.5$ away from UTC^L were marked as incorrect. The following observations emerge from Fig. 5.2:

- Fig. 5.2(a) shows F1 for different values of p and n for $M = 250$ and $t_1 = 0.5$ over all groups in the UTC offset dataset. It can be seen that the best performance with F1 = 68.14% is achieved using $p = 33$ and $n = 5$.

- Fig. 5.2(b) confirms that $p = 33$ is the best performing using precision for four different values of M. This value of p is also an intuitive choice because, as mentioned earlier, about a third of the 24-hour period is expected to be devoted to sleep. Smaller percentile ($p < 30$) reduces the associated sleeping cycle and it is harder to fit a parabola and to get a good UTC offset prediction. On the other hand, if p is too high ($p \geq 40$) then points that are outside of the sleeping cycle will be incorporated causing the performance to suffer.

- Fig. 5.2(c) shows that $n \in [2, 5]$ exhibit high precision for all values of M. From this figure, we conclude that any choice of $n \in [2, 5]$ is reasonable to smooth out irregularities, preserve high precision, but is not too high to delete the sleep cycle from the time distribution. However, considering both, the precision and F1, we conclude that $n = 5$ is the best choice.

- When using sample size M the group size needed to be at least M because we have used sampling without replacement. Sampling with replacement allows to better understand whether improvement comes from a bigger sample size or a bigger group size. Fig. 5.2(d) shows performance for sampling with replacement across different M values as the group size increases (using $n = 5$ and $p = 33$). We conclude that performance is not affected by M, although performance improves with group size.

The plot in Fig. 5.3 uses $M = 250$, $n = 5$, $p = 33$, and group size equal to at least 1000. Using these parameters the relationship between predicted PST and actual UTC is shown. A linear relationship can clearly be observed, using

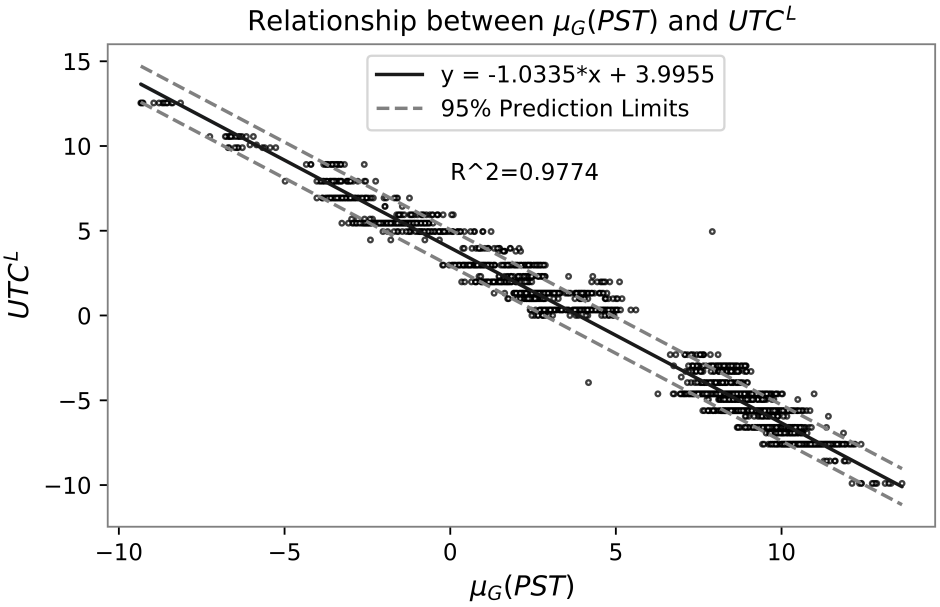


Figure 5.3: Result of linear regression performed on data points with known geolocation, plotting UTC^L against $\mu_G(PST)$, with $M = 250$, $n = 5$, $p = 33$, and group size exceeding 1000.

$UTC^P = -1.0335 \times PST + 3.9955$ with overall $R^2 = 0.9774$; this is approximated as follows:

$$UTC^P = -1.0 \times PST + 4.0 \tag{5.2}$$

5.4 Temporal Analysis of Message Traffic data

In this section, we illustrate that time-based features can be used for associating persons and topics with a geographic area. The time-based approach is confirmed using message traffic with coordinates.

Our focus is on understanding the spatiotemporal aspects of the Twitter social

graph, connecting senders of messages and users mentioned in the messages. We explore the geographical distribution of senders of messages who mention an individual, thereby evaluating the extent to which an influencer (mentioned in the messages) has global influence. This is often accomplished by analyzing message traffic data, since the full follower-followee graph cannot be directly collected due to limitations imposed by the free Twitter API. Messages with coordinates and place mentions or self-reported locations of the users can be used to filter out users that are near a specific geographic area; in this manner, influential individuals and communities belonging to a certain geographic area can be identified.

5.4.1 Message Traffic Dataset

We collected five days of message traffic data in the first week of December 2020 for a total of 18.67 million messages. This dataset is denoted as D_{mess}. Preprocessing consisted of turning each message to lowercase and tokenizing using NLTK library's TweetTokenizer. For each message, the hour was extracted from its creation time. Each token was associated with a set of hours from the set of messages in which the token appears. Tokens that were at least three characters in length and appeared in over 500 messages were retained, resulting in a total of 23,747 tokens.

Messages that contain location coordinates provide ground truth against which we can evaluate UTC-based predictions. Such messages comprised only 0.71% of all messages in our dataset, consistent with other literature suggesting that the number is less than 1% [10]. In our dataset, there were 6,632 messages with point coordinates

Table 5.2: Token labels using messages with geolocation tags

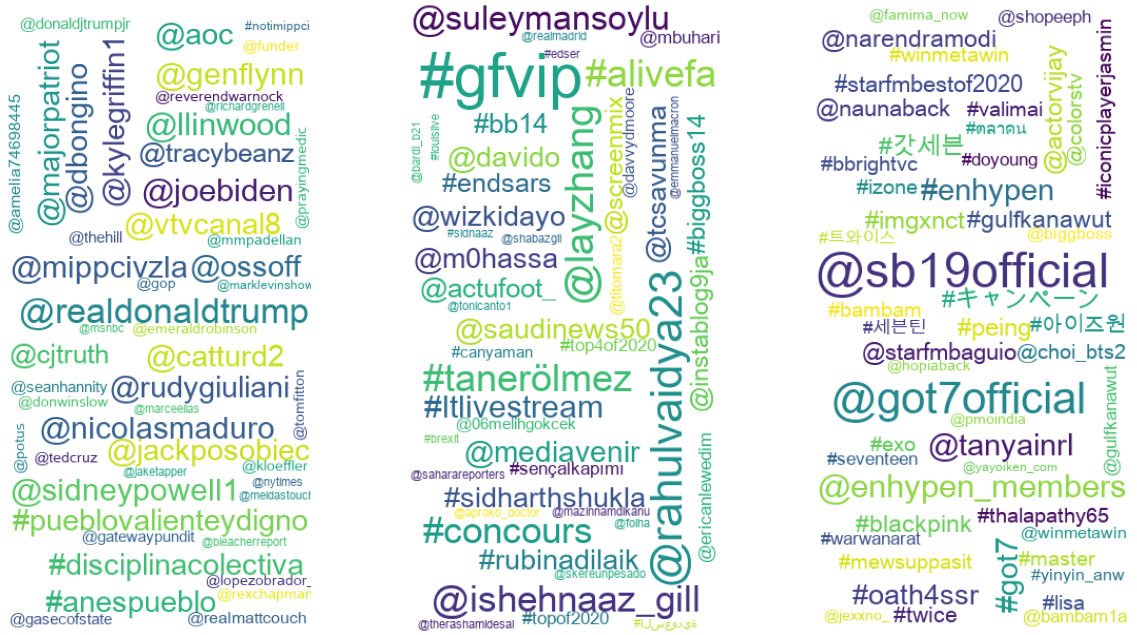
Token	Label	NA_SA	AF_EUR	AS_OC	Total
@realdonaldtrump	NA_SA	537	47	19	603
@joebiden	NA_SA	142	14	6	162
#oath4ssr	AS_OC	18	2	30	50
@narendramodi	AS_OC	1	0	47	48
#gfvip	AF_EUR	0	35	0	35
@pmoindia	AS_OC	0	1	33	34
@thehill	NA_SA	25	2	0	27
@jairbolsonaro	NA_SA	27	0	0	27
@nytimes	NA_SA	21	3	3	27
@llinwood	NA_SA	24	1	1	26

and 126,765 messages with a place coordinate (bounding box).

Among 23747 tokens, as many as 20252 were contained in at least one message with coordinates. For each token, we record the number of messages that came from the Americas (longitude $\leq$ -25), Europe/Africa (-25 $<$ longitude $\leq$ 65), and Asia/Oceania (longitude $>$ 65). For coordinates specified using a bounding box, both the longitude components had to be associated with the same region. A token was assigned a label based on the region which captured the biggest ratio of messages. Among the 20252 tokens with coordinate information, we found that 11955 were associated with the Americas, 4991 with Europe/Africa, and 3306 with Asia/Oceania. Table 5.2 shows examples of ground truth generated in this fashion that contain topic or person mentions (NA_SA = Americas, AF_EUR = Europe/Africa, and AS_OC = Asia/Oceania). The number of messages with coordinates are shown for each region and token. The region that captures the most messages is chosen as the label. For example, for @realdonaldtrump the NA_SA is the label since 537 messages are from it vs. only 47 and 19 for the other regions.

5.4.2 Predicting Region of Token

We extracted the hours (from the creation time) associated with all messages in which each token appears. The set of hours was used to obtain a time distribution and corresponding: (i) predicted UTC offset, (ii) coefficient of the quadratic term, c_2, and (iii) coefficient of determination R^2 (using the approach in Section 3.2). As before, a large value of R^2 implies greater confidence in the fitted polynomial, and a large c_2 indicates greater localization of influence.



(a) Americas (b) Europe/Africa (c) Asia/Oceania

Figure 5.4: Top trending tokens (topics and persons) for the (a) North and South America, (b) Europe and Africa, and (c) Asia and Oceania. These were identified using UTC prediction from time curve over message creation times containing the token.

The NLTK library contains a list of stop-words, such as 'the' and 'has', which are used worldwide. Their temporal distributions are flat and associated c_2 is close

to zero. For example, we found that for stop-words the largest c_2 was smaller than 0.001. To further refine our dataset, we considered $c_2 \geq 0.001$. The result was that not only stop-words but other global topics and persons such as *#covid19* and *@YouTube* were removed.

Out of 23747 tokens in the dataset, 16744 contained a sleep cycle that could be used to predict a UTC offset. Based on predicted UTC offset the token was assigned one of three regions: (i) North and South America ($UTC \leq -2$), (ii) Europe and Africa ($-2 < UTC \leq 4$), and (iii) Asia and Oceania ($UTC > 4$). The number of tokens associated with each region was (i) 9618, (ii) 3012, and (iii) 4114 respectively. Of the UTC predictions, 15087 had $R^2 \geq 0.85$ of which 8487 had $c_2 \geq 0.001$. Among these 8487 higher confidence predictions 4135, 1416, and 2936 belonged to each region, respectively.

As an illustration, Fig. 5.4 shows the top fifty words in a word cloud for each geographic region, focusing on higher confidence tokens that start with # or @ (designating topics or persons).

5.4.3 Evaluation

For each token, one of the three regions using UTC prediction is compared with the ground truth, and the accuracy of prediction is recorded as the ratio of correct versus total predictions, for each region.

Table 5.3 shows the results for the three regions. The first column shows the type

of restrictions placed on a token, based on (i) R^2 of the polynomial over corresponding sleep cycle, (ii) power coefficient c_2 from polynomial, (iii) number of minimum messages, x, used to build ground truth, and (iv) whether a collection is limited to persons/topics (@/#). The accuracy of predictions for each region is shown in columns 3-5 (the respective number of predictions per region is shown in the second column).

Table 5.3: Performance over Message Traffic

Restriction	**Predictions**	**NA_SA**	**AF_EUR**	**AS_OC**
None	9271, 2825, 2602	94.56	76.14	87.78
$R^2 \geq 0.85$	8611, 2426, 2360	95.4	80.3	89.58
$R^2 \geq 0.85$, $c_2 \geq 0.001$	4008, 1327, 1976	98.6	89.9	95.29
$R^2 \geq 0.85$, $c_2 \geq 0.001$, $x \geq 5$	3304, 810, 967	99.21	91.98	98.24
$R^2 \geq 0.85$, $c_2 \geq 0.001$, $x \geq 10$	2270, 380, 533	99.69	94.47	98.69
$R^2 \geq 0.85$, $c_2 \geq 0.001$, @/#	261, 61, 137	98.08	81.97	97.08

Table 5.3 illustrates that the approach using temporal distribution is successful. The first row, with no restriction, illustrates that if a sleep cycle is found and a UTC prediction is made it generally has good accuracy. The accuracy is high, particularly for those tokens which have ground truth assembled from more messages (larger x) and with high confidence UTC predictions (high R^2 and c_2).

About 13% of the tokens that were labeled using UTC did not have any message traffic with coordinates. A bigger collection could be explored, but there is reason to think that some tokens just won't get a geo-tag assigned. Fewer than 1% of messages contained geo-tags, and 38% of the tokens had fewer than 5 geo-tagged data points; a prediction based on such a small sample is not made with high confidence. On the other hand, time is available for all messages and each token appeared in at least 500

messages giving us greater confidence in the corresponding time distribution. This illustrates the usefulness of our approach.

Another alternative would be to utilize the self-reported locations of the users that wrote the messages, but this would require a complex geocoding solution that can handle the different ways persons refer to locations in various languages. Using time distributions is hence a better solution for quickly understanding important keywords in message traffic as they pertain to a geographic region of interest.

5.4.4 Comparison against Baseline based on Google Trends

In a recent paper, Zola et. al. [78] attempt to estimate worldwide Twitter user locations without relying on geolocation target labels (no geotagged tweets or user location profiles and no access to geographic dictionaries). Their dataset consisted of 744,830 tweets written by 3,298 users from 54 countries. The location of each user was manually verified. Their approach focuses on nouns (like sites, events, people), which are expected to have a spatial context that is helpful for user location estimation. Each noun was associated with a geographic region based on Google Trends (Google Trends identifies nouns that are trending in various cities). For each user, clustering is used to identify the most probable centroid from coordinates associated with each city. Because no geoinformation is used, the problem is more complex; their approach correctly predicts the ground truth locations of 15%, 23%, 39%, 58%, 70%, 82% of the users for tolerance distances of 250, 500, 1000, 2000, 4000, and 10000 km. Our method also does not utilize any geoinformation, relying only on creation times as

the feature, and hence it was appropriate to compare our approach against the one based on Google Trends.

In [78] the authors utilize the following approach:

1. Part of Speech Tagging used to identify a set of nouns for each user.
2. Pytrends Python module is used to associate a noun with a list of cities. Google gives each city a weight, from 0 to 100, based on how popular the noun was (based on how many search queries, originating from that city, contained that noun). Cities with scores of zero are given scores of one so that a non-zero value is present for each city.
3. Google geocoder Python module (used to get lat, long of each city)
4. Scikit-learn Python library (used to get the centroid) the best method is based on K-means and Density-Based Spatial Clustering of Applications with Noise (DBSCAN)
5. Centroid is compared to the known location of a user. Median, Mean, and ACC@x (is a user within x kilometers of predicted centroid) are recorded across all users.

We attempt to apply Google Trends to our dataset. In our method we also utilize Pytrends. Pytrend is an unofficial library supporting Google Trends. In function interest_by_region() in file pytrends/request.py we change the code so that the Pandas data frame is returned immediately after collecting JSON response from Google. We find that Google, at the 'City' resolution, does return coordinates for

each city, it is just that Pytrends did not accurately capture this information. In this way, it is not necessary to geocode each city name with steps (2) and (3) combined (this reduces the potential for introducing errors due to additional geocoding).

There are other differences in that we are focused on all tokens (not just nouns) and we already have three predefined regions that the world is broken up into (so the accuracy will be judged based on how well a region is predicted as was done in previous section). Google Trends ranking is used to predict a region for a token using:

1. For each token, we record the set of cities A that came from the Americas (longitude $\leq$ -25), set of cities B that came from Europe/Africa (-25 $<$ longitude $\leq$ 65), and set of cities C that came from Asia/Oceania (longitude $>$ 65).

2. For each set of cities in A, B, C the cumulative score across the cities in each set are recorded. The cumulative score is based on the ranking returned by Google Trends (Google gives each city a weight based on how popular the token was in the city, weight is from 0 to 100).

3. A token is assigned to a region that captured the biggest cumulative score.

Because our problem involves large geographic regions it is also appropriate to utilize the 'Country' resolution vs. only 'City'. Each country is given the average latitude and longitude of its cities.

When using Google Trends by region, it can be used to focus on trends that were formed over a predefined time in the past. These are the predefined time ranges:

Table 5.4: Different time ranges result in very different top 10 country ranking, for keyword @realdonaldtrump, using Google Trends.

1 month	**3 month**	**1 year**	**5 year**	**all**
(KE, 100)	(CA, 100)	(US, 100)	(US, 100)	(US, 100)
(CA, 17)	(NO, 100)	(CA, 84)	(CA, 84)	(CA, 72)
(US, 9)	(IE, 93)	(CN, 35)	(CN, 35)	(IE, 31)
(GB, 4)	(US, 59)	(IE, 34)	(IE, 34)	(SH, 31)
(AF, 0)	(NL, 54)	(NZ, 30)	(NZ, 30)	(NZ, 29)
(AL, 0)	(DE, 22)	(SH, 30)	(SH, 30)	(PR, 23)
(DZ, 0)	(GB, 21)	(AU, 21)	(AU, 21)	(AU, 21)
(AS, 0)	(PL, 17)	(GB, 17)	(GB, 17)	(KE, 19)
(AD, 0)	(IN, 4)	(NO, 16)	(NO, 16)	(CR, 16)
(AO, 0)	(AF, 0)	(KE, 14)	(KE, 14)	(SG, 16)

past 1 hour, 4 hours, day, 7 days, 90 days, 12 months, 5 years, all (2004-present) (Google Trends does not allow one to enter a custom date range i.e. it has to be one of these values). Table 5.4 shows that the trends will result in very different country rankings depending on the time range utilized. The set of country codes in the top three results across the three time frames in Table 5.4 are: KE = Kenya, CA = Canada, US = United States, NO = Norway, IE = Ireland, and CN = China. We choose to focus on 1-year time frame since this is the default option.

The number of requests to Google Trends is limited. For example, Python Pytrends library states that 60 seconds of sleep between requests is recommended in avoiding the limit (we have verified that the limit is around 1440 requests, which greatly reduces the amount of data that can be collected daily). In our evaluation, we have focused on 3183 tokens that had $R^2 \geq 0.85$, $c_2 \geq 0.001$, $x \geq 10$ and 459 tokens that are limited to persons/topics (@/#) (the results for these presented in the previous subsection in second to last and last row, respectively, of Table 5.3).

Table 5.5 and Table 5.6 show the results for the three regions at city and country levels for the 459 tokens and 3183 tokens respectively. The first column shows the type of restrictions placed on Google Trends. Restrictions considered were (i) using only the location with the highest ranking, (ii) using the top three locations, (iii) using locations with a weight ≥ 50, and (iv) using all locations. The last row shows the results using our approach based on message creation times. The second column shows the number of predictions made for each region. The precision of predictions for each region is shown in columns 3-5. The final column is the total number of predictions.

Table 5.5: Performance over 459 Twitter Persons and Topics (@/#)

Restriction	**Predictions**	**NA_SA**	**AF_EUR**	**AS_OC**	**Total**
City using top 1	1, 3, 5	100	100	100	9
City using top 3	1, 3, 5	100	100	100	9
City using weight ≥ 50	1, 3, 5	100	100	100	9
City all	1, 3, 5	100	100	100	9
Country using top 1	161, 28, 52	95.03	92.86	100	241
Country using top 3	161, 28, 52	95.65	92.86	100	241
Country using weight ≥ 50	161, 28, 52	95.03	92.86	100	241
Country all	161, 28, 52	95.03	92.86	100	241
Our Approach using Time	261, 61, 137	98.08	81.97	97.08	459

Table 5.5 shows that, out of 459 tokens, City Google Trends only produced 9 results while Country Google Trends produced 241 rankings. These tokens contained symbols @ and # which on Twitter have special meaning, but these are not as common when using the Google search engine. As a result, Google does not have enough information for trend analysis for these tokens.

Table 5.6 is more informative since the 3183 tokens consist of more common keywords and that tend to be associated with a geographic area. Google Trends has

Table 5.6: Performance over 3183 popular tokens with at least 10 coordinates

Restriction	**Predictions**	**NA_SA**	**AF_EUR**	**AS_OC**	**Total**
City using top 1	2188, 360, 491	78.29	84.72	83.1	3039
City using top 3	2188, 360, 491	80.94	87.5	84.73	3039
City using weight ≥ 50	2188, 360, 491	84.32	90.83	84.11	3039
City all	2188, 360, 491	90.81	94.17	89	3039
Country using top 1	2267, 365, 525	64.49	76.16	88.76	3157
Country using top 3	2267, 365, 525	55.32	75.07	88.38	3157
Country using weight ≥ 50	2267, 365, 525	59.51	78.9	87.62	3157
Country all	2267, 365, 525	56.64	94.79	80.76	3157
Our Approach using Time	2270, 380, 533	99.69	94.47	98.69	3183

information on most of these with 3039 at the city resolution and 3157 at the country resolution. At the city resolution, it is seen that as more cities are considered the precision is gradually going up i.e. performance using just the top city is the worst. On the contrary, the performance using Country Google Trends has better overall performance when using only the top Country. This could be because there are a lot of separate countries in Europe and Africa continent and as a result, this region tends to be heavily favored when using all countries. When looking at precision across all regions our proposed time-based method performs the best with 98.9% overall precision vs. the best results via Google Trends at 69.88% at the country resolution and 90.92% at the city resolution.

Other caveats:

One might need to adjust Google Trends based on population. For example, when resolution is at the country level the keyword 'pizza' is given a weight of 100 for the USA and 99 for Canada. If adjusting for population and number of Twitter users present in the USA vs. Canada, it seems that the USA should be weighted

more. Also, the regions are based on the popularity of search areas where Google is used, but Google and Twitter do not have the same popularity around the world. Google trends seem to be case insensitive i.e. 'day' and 'DAY' both return the same results.

Country Google Trends always returns a ranking for 250 countries (with most countries given a weight equal to zero). When the resolution is at the city level Google will return only the top x cities. For the 3183 tokens the average was $x = 68.78$ +/- one standard deviation of 26.07 (so statistics are not provided for all cities in the world).

City Google Trends associates tokens with city locations with city names and coordinates available. We recorded all unique city names and coordinates, associated by City Google Trends, over 3183 tokens, into set AllCity (5229 cities recorded). Next, each city name from set AllCity was fed to Google Trends and the top city result and its coordinates were recorded. Distance in miles was computed between the coordinates of a city query vs. the top city via Google Trends. As an illustration, Table 5.7 shows example city tokens and the top city association and the distance between the two.

Table 5.7: Example Google Trends top City vs. Known City Query

Query City (coordinates)	**Top Trends City (coordinates)**	**Distance**
Barcelona (41.385064, 2.173404)	Barcelona (41.3850639, 2.1734035)	0
Houston (29.760427, -95.369803)	Bellaire (29.7057858, -95.4588299)	6.5426
Chicago (41.878114, -87.629798)	Norridge (41.9633641, -87.827284)	11.7575
New York (40.712784, -74.005941)	Albany (42.6525793, -73.7562317)	134.4946
Rochester (41.064765, -86.215833)	Rochester (44.0121221, -92.4801989)	378.8369

Across 5229 city queries, 440/5229 = 8.4% produced no results. Out of 4789

queries with results, the average distance between city query and city via Google Trends was 362.03 miles +/- 1334.97. ACC@100 was at 87.57% illustrating that the majority of cities get matched up to a city within 100 miles. However, it is important to highlight that Google Trends is not the same as geocoding i.e. Chicago using geocoder would not get matched up to Norridge, it is just that there were many queries containing Chicago from that location. Similarly, a query such as 'Moscow' will not get associated with Moscow Russia because users there will most likely utilize the Cyrillic alphabet. Cities that are popular travel destinations will also be affected.

In summary, Google Trends is an interesting dataset that could be complementary to the time features proposed. The time features were illustrated to perform better for the proposed task, but Google Trends does have interesting properties. The limitations of Google Trends are related to (i) unavailability of data for certain tokens such as popular Twitter topics/persons, (ii) Google Trends is restricted to about 1400 queries per 24-hour period, and (iii) it is not possible to customize the date range over which trends are formed.

5.5 Evolving Popularity: Inferring Daily Changes in Number of Followers

It is important to understand how an individual's influence changes with time; this can help predict future influence as well. To predict the future one must first understand the past. In the context of Twitter, the corresponding problem involves estimating the rate at which an influencer has gained their existing followers over

a given time period. In this section, we propose a novel algorithm to address this problem, using the creation times of an influencer's followers.

To find the number of followers an influencer has gained on a daily basis (i.e., within a span of 24 hours) during a period of d days, one would need $d+1$ daily collections. Since this is a time-expensive proposition and because Twitter API doesn't allow one to go back in time, we propose an alternative method for approximating daily gains for an influencer, and compare it with an approach based on Meeder, et al. [84].

5.5.1 Dataset: Stable, Global, Growing Influencers

Each Twitter user's profile contains the number of followers that the user currently has. By collecting user's profile multiple times we can get a sense for how the number of followers is changing. Let $\psi(i,t)$ represent the number of followers of influencer i at time t. Let $\psi(i,t_0,t_1) = \psi(i,t_1) - \psi(i,t_0)$ represents the number of new followers i gains during the time interval $[t_0,t_1]$; the number of followers stated in influencer's profile at t_1 minus the number of followers stated in influencer's profile at t_0.

Twitter keeps track of popular influencers via *@verified*. There are over 300K verified influencers as of this writing. Our focus is on global stable verified influencers that continue to gain followers; to this end, we collected data on influencers that met the following criteria. The influencer:

1. having greater than a million existing followers, i.e., $\psi(i,t) > 10^6$;

2. gaining at least a thousand followers within 24 hours,i.e., $\psi(i, t_0, t_1) \geq 10^3$ where t_0 and t_1 are 24 hours apart;

3. the gain in the number of followers is less than 1% of the overall existing follower base, i.e., $\psi(i, t_0, t_1) \leq (0.01 \times \psi(i, t_0))$;

We ensured that the above criteria were met over three $\psi(i, t_0, t_1)$ collections performed in December 2020. Let U_0 contain all verified Twitter influencers. For each influencer i in U_0 we computed $\psi(i, t_0 = d_0, t_1 = d_1)$ where d_0 and d_1 are 24-hours apart. Influencers that met the three criteria from above form the set U_1. For each influencer i in set U_1 we ensured that the three criteria were again met using $\psi(i, t_0 = d_2, t_1 = d_3)$ where d_2 and d_3 are 24-hours apart to obtain set U_2. The process is repeated again using $\psi(i, t_0 = d_4, t_1 = d_5)$ yielding the final set U_3 consisting of 600 influencers.

Data collected for each Influencer

The data collected is used to illustrate that $\psi(i, t_0, t_1)$'s, where t_0 and t_1 are 24 hours apart, can be predicted using the creation times. Using an instance of the Twitter API we collected the first 50K followers for each influencer $i \in U_3$. Twitter API instance is used to record the profile metadata and store them to $allProfile = \{allprofile(t, i) : i \in U_3, t \text{ refers to the time of collection}\}$. Profile collection is repeated every 5 minutes with the list of collection times given by PC.

Another Twitter API instance collects followers. The follower collection, unlike profile metadata, cannot be performed quickly across all influencers. The time when

influencer i's followers are collected is recorded as Followers$_t(i)$.

Once the followers for all influencers are collected: given an influencer i, PC is used to find a the closest time to Followers$_t(i)$ (which we refer to as t_1) and to Followers$_t(i)-24$ hours (which we refer to as t_0). Recall $\psi(i, t_0, t_1) = \psi(i, t_1) - \psi(i, t_0)$, in this case $\psi(i, t_0, t_1)$=$allProfile(t_1, i)$-$allProfile(t_0, i)$.

In this way, we have $\psi(i, t_0, t_1)$ over the same period that the followers were collected for all users in U_3. In the rest of the chapter, we refer to $\psi(i, t_0, t_1)$ over all users in U_3 as the actual 24 hour follower gain, a_{24}.

The followers and the a_{24} over all users in U_3 make up our dataset that is denoted as D_{600}. Table 5.8 shows ten influencers from our dataset ordered by the highest a_{24}.

Table 5.8: Follower gain for selected influencers over 24 hours

Influencer	**Follower at t_0**	**Follower at t_1**	**a24=Gain**
joebiden	22009684	22057780	48096
bts_twt	31718727	31766383	47656
bts_bighit	26238967	26280220	41253
arianagrande	80458070	80494729	36659
elonmusk	41178206	41208685	30479
bighitent	18527632	18553608	25976
kamalaharris	13553348	13577323	23975
narendramodi	64532998	64556393	23395
iamcardib	15913538	15935631	22093
nasa	42743031	42763315	20284

5.5.2 An Algorithm to Estimate Follower Gain

Meeder et al. [84] observed that the followers of an influencer are returned by Twitter in a list that is in the order of following time i.e. most recent follower first.

Dataset D_{600} for each influencer contains 50K followers. For a specific influencer, let $\mathcal{L} = [l_0, l_1, l_2, \ldots, l_{49999}]$ be the list of creation times of its followers. We select the first $24 \times n$ values from this list for generating 24 rows of size n each, denoted as $L_1, L_2, \ldots, L_{24}$. Each L_i is used to generate a time distribution of the creation times. For example, in Fig. 5.5, we have plotted 24 such distributions, using $n = 600$ for the influencer *@CNN*. In this figure, for each distribution, the hour during which the frequency peaks is highlighted in red. We observe that each distribution has a peak and the peak shifts by an hour. Fig. 5.5 is drawn for *@CNN* but a similar behavior is observed for most global influencers. In the following, we describe the novel algorithm to estimate an influencer's daily follower gains.

Representing Cyclical Nature of Time

In the 24-hour clock, shown in Fig. 5.6, each hour can be represented using sine and cosine values of the angle the hour-hand makes with the vertical straight line from the center to the 24^{th} hour. The angle for an hour h in degrees is given as $\frac{360 \times h}{24}$. For example, the angle for 2 O'clock is $\frac{360 \times 2}{24} = 30^o$ and 2 O'clock is represented as $(\sin 30^o, \cos 30^o)$ = (0.5, 0.866). Similarly 5 O'clock is expressed as $(\sin 75^o, \cos 75^o)$ = (0.965, 0.258). The cosine similarity between (0.5, 0.866) and (0.965, 0.258) is 0.707. If we plot the cosine similarities of (sine, cosine) representation of a specific hour A, with (sine, cosine) representations of hours A, (A+1), (A+2), ... , we obtain a smooth cosine curve (see red plot on the right of Fig. 5.6). The vector of the above cosine similarities is denoted as V_1 and is called the optimal vector.

Now consider the peak hour in each distribution of *@CNN* in Fig. 5.5. If we

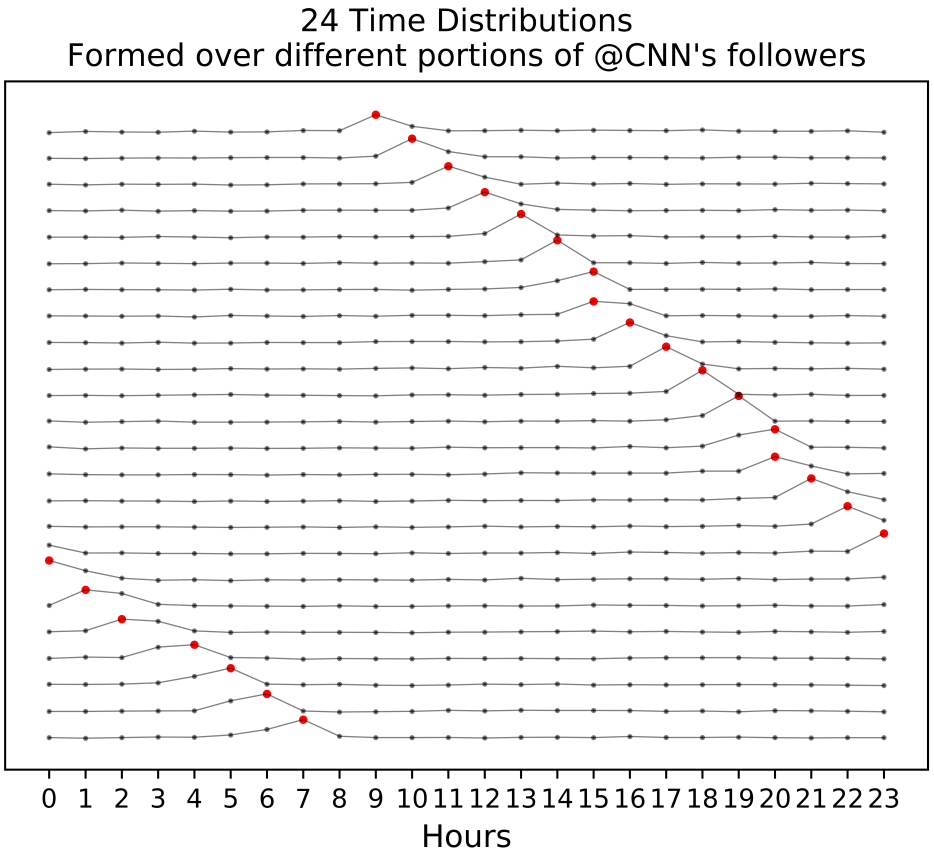


Figure 5.5: 24 time distributions where each time distribution formed from $n = 600$ followers of *@CNN* for a total of $24 \times 600 = 14400$ followers. Distributions are plotted one above the other $(L_1, L_2, \ldots, L_{24})$. For each distribution the hour during which it peaks is highlighted in red.

compute and plot the vector V_2 of cosine similarities between (sine, cosine) representations of these hours with the representation of hour 7 (where the peak occurs in the first distribution), we obtain the brown curve in Fig. 5.6. Likewise, vectors of cosine similarities resulting from sample sizes given by $n = 100$, 200, 300, 400, and 500 are shown. The similarity between V_1 and V_2 can be computed using ρ, the Pearson Correlation Coefficient.

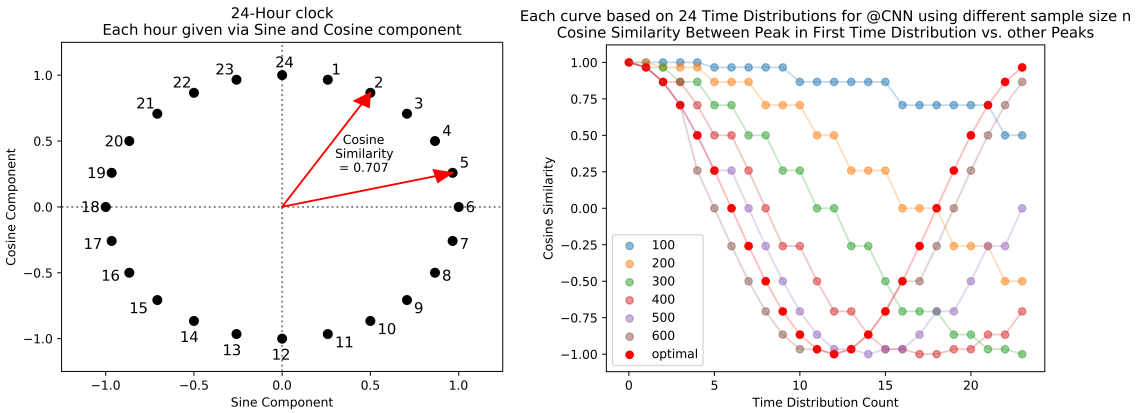


Figure 5.6: **Left** – 24-hour clock; illustrates the computation of the cosine similarity between two hours. **Right** – Each curve is computed using cosine similarity between the peak time in the first time distribution vs. itself and peak times over remaining 23 time distributions for various n as described in the text (all graphs for *@CNN*). The red curve is optimal in that the peaks are in the order of the hours on the clock.

We are interested in the size of n that results in temporal distributions that peak exactly one hour apart for all 24 hours (or as close to it as possible). For example, in Fig. 5.6, the curve associated with $n = 600$ is closest to the ideal red curve. The key idea is to try different values of n and calculate the associated V_2 vectors. The vector V_2^* with the highest correlation against V_1 and associated n^* are obtained. The number of followers gained over 24 hours is predicted as $p_{24} = 24 \times n^*$. A formal description of the algorithm is provided below.

The Algorithm

Input to the algorithm is the list, $\mathcal{L}$, of an influencer's followers and the precomputed optimal vector V_1. Next, n is chosen from a minimum of 10 to a maximum of $\lfloor \frac{|L_1|}{24} \rfloor$. For each n, 24 time distributions are generated and from each, the hour

during which the time distribution peaks is recorded. V_2 is generated using cosine similarity between the peak hour in first time distribution vs. peaks across all 24 time distributions. V_1 and V_2 are compared using the Pearson Correlation Coefficient ρ. The sample size n^* that resulted in the highest correlation coefficient is returned. The predicted 24 hour followers turn over, p_{24}, is given as $24 \times n^*$.

Algorithm 1: *infer24HF(L1)*:

```
  Input: List L1 of follower creation times;
  Output: Predicted 24 Hour Follower Gain, associated
  Pearson Correlation, and number of unique peaks;
  bestN, maxP, maxH = 0, 0, 0;
  V1 = vector of cosine similarities between
      hour 0 and hours [0, 1, 2, ..., 23];
  for n in [10, 15, ..., |L1|/24]:
    Split first 24*n elements of L1 into 24 bins of size n;
    Record the hour with most elements for each of 24 bins;
    V2 = vector of cosine similarities between
        hour in bin 1 and hours in each bin;
    P = Pearson Correlation between V1 and V2;
    if P > maxP:
      bestN = n;
      maxP = P;
      maxH = number of unique peaks across bins;
  Return bestN*24, maxP, maxH;
```

```
end
```

5.5.3 Evaluation

For each influencer in D_{600}, we compute p_{24} and compare it to known follower gain a_{24}, using the comparison measure *diff*(p_{24}, a_{24}) = $\max(p_{24}/a_{24}, a_{24}/p_{24})$-1.

Fig. 5.7 shows the scatter plot of p_{24} versus a_{24} for all influencers in D_{600}. The scatter plot is color-coded: green dots represent influencers with *diff* ≤ 0.25, and red dots represent large differences with *diff* > 1.0. The Pearson correlation coefficient between p_{24} and a_{24} vectors is $\rho = 0.967$, a high value that shows that the proposed method makes accurate predictions.

We compare our algorithm against two baselines. Meeder et al. [84] provide a method for estimating when a user had followed the influencer. Given a list of followers' creation times L_1 for influencer i, the follow time for a follower at index j is approximated by $\max(L_1[j:])$ (max gives the most recent creation time at indices greater than or equal to j). For our problem we are interested in the number of followers gained over 24 hours so that the datetime $\max(L_1[j:])$ is as close to the datetime that is 24 hours before the follower collection took place (given by $\text{Followers}_t(i)$ minus 24 hours). Effectively we are trying to utilize the method proposed by Meeder to estimate the index j that would satisfy this requirement. The method should work for those influencers that are likely to be followed by brand new users immediately after their account creation.

$\text{Followers}_t(i)$ gives time t for influencer i's followers collection. Let $\mathcal{L}_M[j] =$

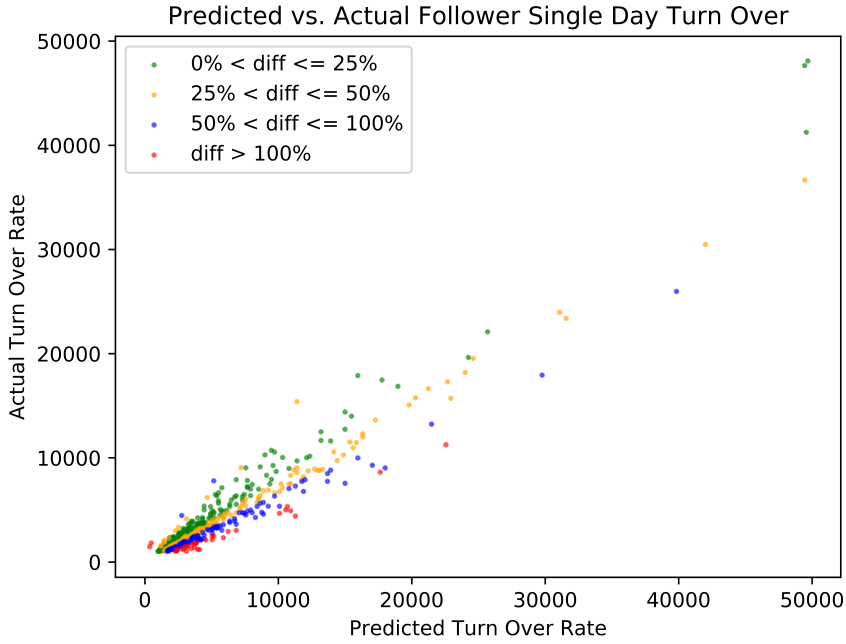


Figure 5.7: Scatter plot of Inferred vs. Actual number of followers gained by 600 influencers over a 24-hour time period (Pearson correlation coefficient = 0.967); 238 points (green) differ by $< 25\%$, 195 points (orange) differ by $< 50\%$, 126 points (blue) differ by $< 100\%$, and 41 points (red) differ by $\geq 100\%$.

($t -$ creation time of the j^{th} follower of the influencer, for $j = 0, 1, \dots$).

Baseline 1:

Traverse the list, $\mathcal{L}_M$, in reverse order and find the first index j, such that $\mathcal{L}_M[j] \leq$ 24 hours. If such a j exists, then return $j + 1$, denoted as $p_{24}^{(B1)}$; else return $|\mathcal{L}_M|$.

Baseline 2:

For each j, such that $\mathcal{L}_M[j] \geq$ 24 hours calculate $\frac{\mathcal{L}_M[j]}{j+1}$; Find the minimum ratio, which will approximate the average number of seconds that elapse per new follower;

Return $p_{24}^{(B2)} = \frac{86400}{ratio}$ (since there are 86400 seconds in 24 hours).

As before, we can calculate the correlation coefficient between the vectors of $p_{24}^{(B1)}$ and a_{24} and between $p_{24}^{(B2)}$ and a_{24} over all influencers. In addition, median error and MSE can be computed, where diff(predictions, a_{24}) is the error that is to be minimized. Table 5.9 shows how our approach compares against baseline predictions based on these measures. Correlation values of all three approaches are high, with slightly better values obtained by our approach. In terms of median error and MSE, our approach performs much better than the baselines.

Table 5.9: Performance of three algorithms to predict influencers' gains

Approach	**Correlation**	**Median Error**	**MSE**
Baseline 1 ($p_{24}^{(B1)}$)	0.962	0.620	0.665
Baseline 2 ($p_{24}^{(B2)}$)	0.964	0.541	0.510
Our	0.967	0.298	0.252

5.5.4 Rationale for Proposed Algorithm and its Limitations

If we consider a group of users that acted during a specific hour h (such as posting a message or following another), then we are likely to observe a maximum near that same hour in their creation time distribution. This behavior has been confirmed, as discussed below, by analyzing time distribution for users grouped using the time that they have posted a message.

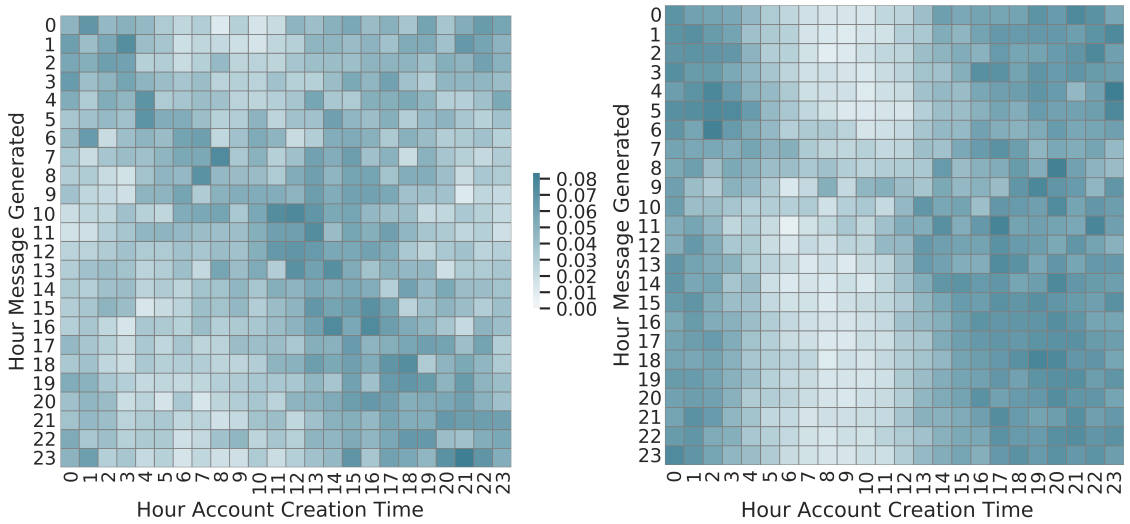
We utilize the dataset D_{mess}. We take all messages that contain a specific token. For example, for token '@youtube' there were 13704 messages. Next, we separate the messages (containing that token) by the hour of message creation time. In this way,

24 groups of users are formed where each user group is known to have been active during a specific hour (the hour during which the message was generated). For each user group, we construct the creation time distribution.

Fig. 5.8(a) shows a heat map for the 24 time distributions generated for token '@youtube'. Notice that a global concept '@youtube' will have a pattern down the diagonal like an Identity Matrix ('@youtube' considered global because $c_2 < 0.001$); the same analysis was performed using stopwords such as 'the' and 'you' and they also observe this pattern. The pattern is due to a unimodal distribution that peaks near the same hour as the hour during which the users were most active in generating the messages. Intuitively if a person had the time for creating their Twitter account in the morning then this person is likely to be active on the Twitter platform during the same morning hours in the future (there is thus a correlation between the creation times and activity times).

The distinction between global and local influencers is illustrated by comparing Fig. 5.8(a) vs. Fig. 5.8(b). Fig. 5.8(b) focuses on a more localized token 'trump' that clearly has a period of inactivity, a sleep cycle, during hours 5-11 (token has $c_2 > 0.001$ and during the collection period it was heavily discussed in the Americas).

The concepts observed over message analysis apply to studying the influencer's followers. We do not know when a user followed an influencer, but because the followers are in sequence of follow time this indicates which followers must have followed earlier on. Algorithm 1 attempts to find a batch of followers of size n that results in a unimodal distribution, which indicates that the followers are likely to have followed the influencer during the same hour as their account creation. When



(a) Messages with '@youtube' (b) Messages with 'trump'

Figure 5.8: Heat maps showing 24 time-distributions from users' creation times where users are binned by the hour that they generated messages containing tokens: (a) '@youtube' (global) and (b) 'trump' (local). For a global token like '@youtube' we see that if a user was active in posting a message during hour h, then the user was likely to have created their account near the same hour h. For token 'trump' a sleep cycle is observed (period of inactivity hours 5-11).

24 batches, of size n, each peak during a different hour in sequence, it gives confidence that the follower gain around the 24-hour time period has been accurately identified (as has been illustrated in Fig. 5.5 for followers of *@CNN* and like the Identity Matrix in Fig. 5.8(a)).

Algorithm 1, for this reason, is well suited for global influencers that are gaining followers around the clock. In contrast, the heat maps for localized influencers show no strong peaks during some hours of the day. The approach, presented above, also cannot be relied upon for influencers that are gaining no more than 50 followers a day, because the average hourly batch will be too small to generate a meaningful time distribution.

There will be periods during which an influencer gains no followers and even loses followers. We can reason only about followers that the influencer currently has, i.e., we cannot know which followers an influencer might have had in the past. If the influencer has lost many original followers, then the signal in the data will be obscured by considerable noise; ρ will be small since the peaks will not cover all hours, and the order might not be perfect. Hence we have chosen to focus on influencers that have a large stable following and that are continuing to increase their follower base. It is preferable to pay close attention to ρ and to stop making inferences after ρ goes below some threshold. It is also recommended to compare the modified baseline based on Meeder et al. [84] as an additional check against our method.

5.5.5 Studying the Evolution of Popularity

To study an influencer's evolution of popularity we need to find how many followers the influencer has gained over multiple days. In an earlier subsection, we have shown that we can estimate an influencer's follower gain over past 24 hours. The same technique can be repeatedly applied to study the gains over a longer time span.

To understand the evolution of an influencer's popularity, we first find its followers' creation time list, L_t, obtained at time t. Unlike the list in the previous section that contained only 50K followers, this list consists of all available followers of the influencer.

Say we have an influencer with 10 million followers. We could send the whole list to Algorithm 1, but it is not reasonable for the influencer to have gained 10 million followers in 1 day, and so to reduce computation we send a smaller more reasonable list. The feature $wSize$ sets the threshold for the maximum number of followers to send to Algorithm 1 (this threshold can be increased or decreased based on influencer's popularity).

Using the first $wSize$ followers between indices [0, $wSize - 1$] of L_t, Algorithm 1 calculates the number of followers gained between t and $t-1$, denoted as p_{24}^t. The next $wSize$ followers between indices: [p_{24}^t, $wSize + p_{24}^t - 1$] will calculate p_{24}^{t-1} (gain between $t-1$ and $t-2$). The next $wSize$ followers between indices: [$p_{24}^t + p_{24}^{t-1}$, $wSize + p_{24}^t + p_{24}^{t-1} - 1$] will calculate p_{24}^{t-2} (gain between $t-2$ and $t-3$). The daily gains returned as list: [p_{24}^t, p_{24}^{t-1}, p_{24}^{t-2}, ...] successively going backward in time.

Using this approach with $wSize = 50000$, Table 5.10 illustrates the number of

followers gained in the last 10 days by two examples of qualitatively different kinds of influencers: *@MrBeastYT* and *@NPR*. The table also contains the associated correlation values (suggesting the degree of confidence), and the maximum number of unique hours captured by the peaks for each calculation from Algorithm 1. We observe that *@MrBeastYT* consistently adds more followers than *@NPR*. *@MrBeastYT* also has higher unique hours and higher correlation, suggesting greater confidence in these predictions. This is reasonable since a more popular influencer will have more hourly followers, and consequently, the time distribution will be formed using more data points.

Table 5.10: Comparison of numbers of followers gained ($p_{24}{}^{\text{t-d}}$) over each of 10 days by two influencers, along with correlation values maxP and the number of hours maxH spanned by the followers in each 24-hour period

	@MrBeastYT			***@NPR***		
period t-d	$\mathbf{p_{24}{}^{t\text{-}d}}$	**maxP**	**maxH**	$\mathbf{p_{24}{}^{t\text{-}d}}$	**maxP**	**maxH**
t-1	12480	0.989	20	1680	0.759	18
t-2	12120	0.993	21	1560	0.944	18
t-3	14400	0.98	22	1440	0.678	20
t-4	9480	0.98	22	1800	0.847	16
t-5	10800	0.989	23	1440	0.91	20
t-6	12960	0.934	20	1320	0.944	17
t-7	10800	0.981	21	1560	0.834	19
t-8	11520	0.978	22	1680	0.972	20
t-9	11520	0.979	21	1440	0.948	21
t-10	10440	0.984	22	1560	0.967	21

The evolution of popularity for these two influencers can be visualized using Fig. 5.9, generated by repeatedly taking $n = 200$ followers at a time. The x value corresponds to the index of the last follower in the sample $[n, 2n, 3n, \dots]$. Time distribution is formed over followers using indices $[x - n : x]$ and the hour during which time distribution peaks is recorded. The cosine similarity between the first

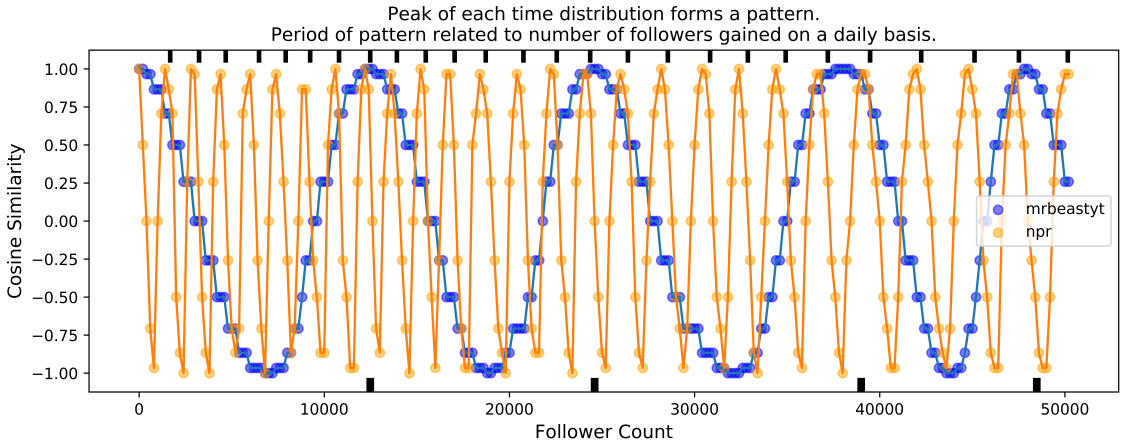


Figure 5.9: Daily follower gains from the proposed method are shown as black tick lines on top for *@NPR* and on the bottom for *@MrBeastYT*. The cosine similarity curve, as described in text, has a periodicity that predictions from the proposed method can capture. We can thus visually verify that the proposed method is making meaningful predictions going backwards in time beyond a single day.

peak hour vs. the sequence of all peak hours is recorded.

The cosine similarity curve has a periodicity (it starts at 1 goes to -1 and then back to 1). The predicted p_{24}^{t-d} from Table 5.10 are shown using black tick lines at the top of the chart for *@NPR* and the bottom for *@MrBeastYT*. For example for *@MrBeastYT* the black tick lines appear at $[p_{24}^{t} = 12480,\ p_{24}^{t} + p_{24}^{t-1} = 24600,\ p_{24}^{t} + p_{24}^{t-1} + p_{24}^{t-2} = 39000,\ \ldots]$. Visually we can see that the black ticks correspond to the periodicity of the curve for each influencer. In this way, another way to think about our method is in being able to capture the lengths of the periods in Fig. 5.9, which happen to correspond to the past number of daily followers gained.

5.6 Global vs. Local Influencer Classifier

In this section, we consider the problem of classifying local versus global influencers. For this, we generate a labeled dataset with 680 local and global influencers. The features are based on sleep cycle analysis (from Section 3.2) and peak analysis (from section 5.2). The resulting classifier illustrates that the features proposed in this chapter are well suited for the task.

5.6.1 Dataset

The method from [83] is used to generate a list of global and local influencers. Automated Google search queries are utilized to get top Twitter influencers associated with the 100 most populous US cities. The followers of the top influencers are used to generate communities representative of each city. A modified TF-IDF algorithm is used to rank influencers based on whether they have a strong connection to a single city community (local) vs. multiple communities (global). Each influencer was verified manually by reading the influencer's description and other profile meta-data. In this manner, 680 influencers were identified out of which 558 were local and 122 were global.

5.6.2 Features

Given a new influencer, we collect the list L_t, of up to 50K followers. Next, Algorithm 1 is applied over L_t to generate features: $p24$, $maxP$, and $maxH$ (F_0 to F_2 listed

below). In the following, the temporal distribution, resulting from the first $p24$ followers in L_t is denoted as $p24Dist$.

1. $F_0 = p_{24}$; if $p_{24} < 500$, $p_{24} = 500$.

2. $F_1 = maxP$: the associated ρ.

3. $F_2 = maxH$; the maximum number of unique hours with peaks.

4. A quadratic is fitted over sleep cycle in $p24Dist$ (as described in section 3.2):

$$F_3 = \begin{cases} c_2 & \text{if sleep cycle exists and quadratic is parabolic} \\ 0 & \text{otherwise.} \end{cases}$$

5. $F_4 = std(p24Dist)$, the standard deviation associated with $p24Dist$.

6. F_5= the fifth Fourier Coefficient (we tried the top 10 Fourier Coefficients[3] associated with $p24Dist$, but the final classifier did not find others significant. A time distribution with a quadratic will need to be represented using higher order Fourier Coefficients, $F_5 > 0$. Conversely, a simple linear function can be represented using fewer coefficients so that $F_5 == 0$.

5.6.3 Results – Local versus Global Classification

We use four families of classifiers:

1. Support Vector Machine (SVM) with the dot, radial, and polynomial kernels,

[3]Complex Fourier transform was used with the SciPy mathematical Python library. The real coefficients corresponding to the cosine terms recorded.

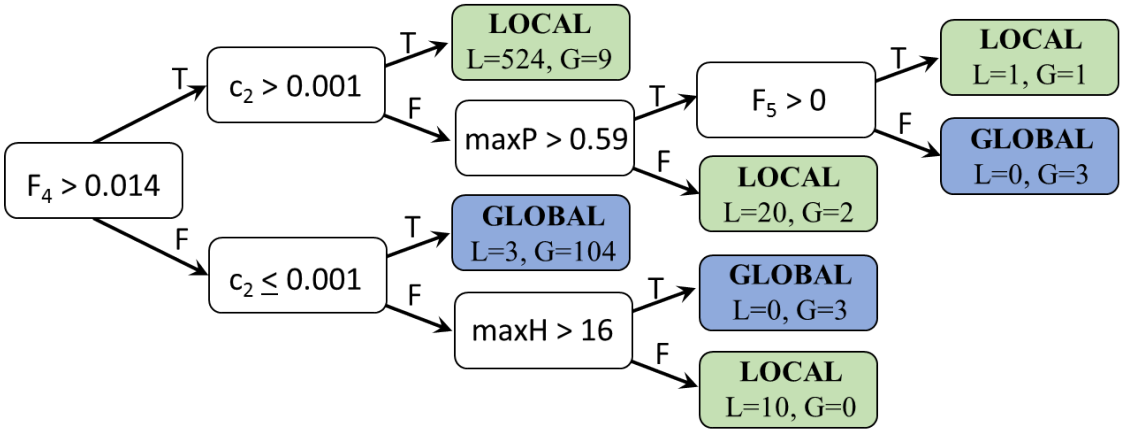


Figure 5.10: Decision Tree Classifier for differentiating local vs. global influencers based on the features from account creation times of their followers. The number of local (L) and global (G) influencers predicted using each branch shown for each leaf node.

2. Naïve Bayes,

3. Decision Tree; using information gain with max depth = 5, and

4. Random Forest; the number of trees ≤ 10, each tree uses information gain with max depth = 5.

Cross-validation with $K = 5$ was employed. Accuracy is averaged over 5 iterations. Decision Tree gave the best results with an average accuracy of $(96.91 \pm 1.08)\%$, followed by the Random Forest $(96.18 \pm 0.86)\%$, and Naïve Bayes $(96.18 \pm 1.27)\%$; SVM performed poorly for all three kernels. The Decision Tree Classifier is shown in Fig. 5.10.

We used information gain to rank the features. Top four features and their associated weights are:(i) F_1: 1, (ii) F_4: 0.983, (iii) F_2: 0.972, (iv) F_3: 0.956 (the weight for F_5: 0.058 so it is not as significant).

As we have seen in the previous section, a sample of followers from a global influencer can lead to a time distribution that is unimodal, and for this reason, it is important to take a sample determined by Algorithm 1. Algorithm 1 searches for the optimal curve that is achieved if the peaks from time distributions are in sequence and contain all 24 hours; if the ρ (F_1) is low and if a small number of hours (F_2) are covered this indicates a local influencer.

If F_4 ($std(p24Dist)$) is low then the spatial distribution is flat and belongs to a global influencer; which is consistent with observations made in the previous sections. The information gain identified that F_3 (c_2) less than 0.001 should be the cutoff for a global influencer (this exact value was also confirmed from analysis of stop words using message traffic in Section 5.4). Finally, if the time distribution is represented using only low order Fourier coefficients so that $F_5 = 0$ this means this is more of a flat line simple time distribution associated with a global influencer.

This classifier is intuitive and over the whole dataset achieves $665/680 = 97.79\%$ accuracy. The followers of influencers that the classifier predicts as local can be used for predicting UTC offset related to local expert finding in social networks. While the followers of global influencers can be used for inferring daily follower gains and analyzing how their popularity has evolved.

5.7 Conclusions

In this chapter, we have illustrated an approach for how creation times can be used in time series analysis. The creation times can stem from a group of messages or account

creation times. It was illustrated that the distribution of creation times that stem from a single time zone will be approximately parabolic, with a minimum during the night time for that time zone. Regression with a quadratic function can be used to predict the UTC offset associated with the time zone. By examining message traffic, this information was utilized to identify trending keywords over multiple geographic areas of interest. In addition, by analyzing the set of followers of any influencer, we showed that this information can be utilized to determine how strongly localized is the range of influence of an influencer. This is useful for Location-Aware Influence Maximization (LAIM) and local expert finding in social networks.

We also illustrated that a follower sample exists such that the peaks from multiple time curves occur in sequence. Analysis of variations of the wave pattern in the distribution of peaks provides information regarding the periodicity with which followers were gained. This is useful for understanding how an influencer's popularity has evolved over time, as well as for inferring link creation times.

Finally, the proposed time-based features were utilized for creating a local vs. global type classifier. The classifier is important because the UTC offset prediction should be applied for local influencers whereas the analysis for how influencer's popularity evolved works for global influencers.

Chapter 6

Application 1: Multilingual Geocoder based on labels using Time and Language Features

6.1 Introduction

As has been shown in Chapter 2, it is inherently difficult to build a universal geocoding solution that can handle various languages and their associated alphabets, popular slang, and purposely ambiguous phrases on Twitter. In a step towards a universal geocoding solution, Chapter 5 showed that time-based features could be used for identifying whether the user group is from a particular timezone. A timezone spans a large geographic area, but the language features can constrain the set of possible countries. In this chapter, the proposed approach is illustrated by categorizing 320K Twitter influencers. High confidence influencer predictions are used as training data for an improved geocoder. This geocoder automatically learns popular ways that Twitter users refer to locations within the country and can handle foreign alphabets.

6.2 Influencers-Dataset

In this dataset, user groups are binned by the influencer they follow. The influencers for the dataset were chosen from the special Twitter *@verified*. It tracks all influencers

that have passed an internal Twitter check (after Twitter performs a special check the influencer is identified via a special blue badge). In this dataset, collected in the spring of 2019, there are 320,166 influencers. Due to Twitter API limits, for each influencer only a single API call was made which returned at most 5000 followers.

The ground truth consists of the country associated with the self-reported location reported by the influencer. It is checked whether this country can be used as a label, based on whether this country matches (i) the most frequent country from self-reported locations of followers and (ii) whether it is contained within the set of countries that would be predicted using followers' time and language features. It is shown that time and language features can be used to improve the precision of the country labels. The country label and the associated influencer's followers' self-reported locations are used for training and illustrating a multilingual geocoder.

6.3 Incorporating Language

We utilize language to further improve the performance of the time-based classifier proposed in section 5.3. Given a time distribution associated with a user group we can compute UTC^P (equation (5.2)). Let U_1 equal the set of countries whose cities have a time zone that observes UTC offset in the range $[UTC^P - t_1, UTC^P + t_1]$, where t_1 is a preselected threshold. For example, the set of countries [TLS, PLW, JPN, MNP, FSM, GUM, IDN, AUS, PRK, RUS, KOR, PNG], correspond to UTC range $\in$ [8, 10].

Our next step is to incorporate language information to constrain the set of possible countries for the selected UTC offset range. Initially, the CIA World Factbook was utilized for this purpose. But for some countries, the languages from CIA World Factbook did not align with the languages used on Twitter. For example, English is the most popular language in India for communication, but it does not make it into popularly spoken languages (instead Hindi, Bengali, and others are listed).

We utilized language preferences, a user-selected option, which is available for more than 99% of collected users on Twitter. In our dataset, there are 76 unique language codes associated with 143 countries, each language was used by at least 100 users, collected over 373 million user profiles.

Given a user group, the users' language preferences are used to generate a language distribution, i.e., we calculate:

$$g_G(\ell) = \frac{\#\text{ of users of language } \ell \text{ in } G}{|G|}$$

for all 76 languages. The language distribution for all countries was also calculated, i.e.,

$$h_c(\ell) = \frac{\#\text{ of users of language } \ell \text{ in country } c}{\#\text{ of all users in country } c}$$

for all 143 countries.

Using cosine similarity $g_G(\ell)$ is compared with $h_c(\ell)$ for all 143 countries. Let U_2 be the set of countries whose language distributions have a similarity score greater than threshold t_2. Finally, let $U_3 = U_1 \cap U_2$ that is, U_3 equals the set of countries common to both sets U_1 and U_2.

Table 6.1 shows how the language distributions helps narrow down the list of possible countries for different thresholds t_2. Recall that for each user group in the UTC offset dataset its location and therefore its country is known. For each user, if this country is in set U_i then the prediction is marked correct, incorrect otherwise. In Table 6.1 the first three columns are the median number of countries associated with U_1, U_2, and U_3.

Table 6.1: Language helps constrain the set of possible countries from UTC while improving precision for higher cosine similarity, t_2

$\mathbf{U_1}$	$\mathbf{U_2}$	$\mathbf{U_3}$	**P**	**R**	$\mathbf{t_2}$
19	143	17	89.02	100.00	0.05
19	140	17	89.14	99.87	0.15
19	127	12	89.54	99.42	0.25
19	115	8	89.75	99.18	0.35
19	108	7	90.07	98.80	0.45
19	102	7	90.23	98.58	0.55
19	99	6	90.32	98.38	0.65
19	91	6	90.27	98.23	0.75
19	83	5	90.26	97.84	0.85
19	70	4	90.34	96.01	0.95

It can be seen that our final prediction, identified by the set U_3, has good accuracy and a much smaller set of possible countries then initial U_1. From table, values from 0.85 to 0.95 work well for t_2.

6.4 Illustration over Influencers-Dataset

The prediction methods, discussed in previous sections, are shown in a pipeline in Fig. 6.1. As an application, the pipeline is applied over Influencers-Dataset. The goal is to identify geo-influencers and accurately associate them with the country

that most of their followers are from. We are interested in high confidence influencer predictions because these in turn can be used as training data for a multilingual geocoder that will be presented in the following section.

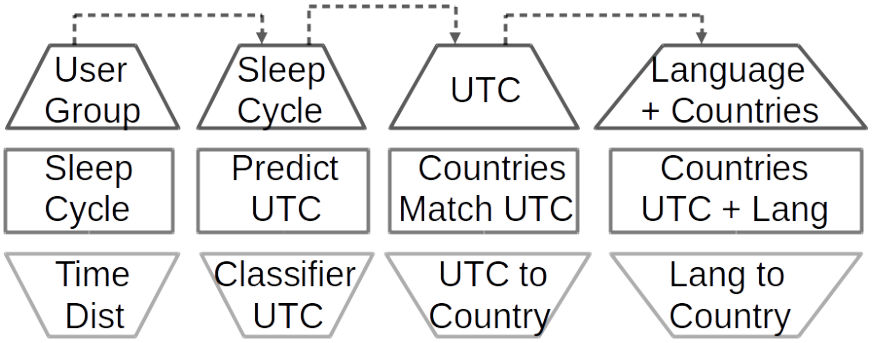


Figure 6.1: Pipeline for predicting region of the world using time and language features. The top layer shows input, the middle layer output, and the bottom layer the name of the process. Each step sends its output as input to the next step.

One data point is the influencer's self-reported location. Another data point is the ratio of followers' self-reported locations belonging to a particular country. The expectation is that for geo-influencers the country associated with the self-reported location will match the most frequent country from influencer's followers. Self-reported locations that match a known GeoNames entry are utilized as described in section 5.3.1.

A problem with self-reported location information from followers is that the labeled location focuses on the Latin alphabet with foreign city names appearing as they would be referred to by an English speaker (Moscow instead of Moskva in Cyrillic). Therefore, English-speaking users are incorporated more often than non-English speakers. Using the following steps we aim to test the different stages from Fig. 6.1 to help with this imperfect baseline:

1. Step 1a (S1a) – Get rid of influencers whose followers' time distribution does not exhibit a U-shaped parabola.

2. Step 1b (S1b) – Get rid of those influencers which are classified as global.

3. Step 2 (S2_t_1) – UTC^P computed and used to obtain the set of possible countries U_1.

4. Step 3 (S3_t_2) – Reduce the set U_1 to set U_3 by constraining to countries that have cosine similarity above t_2.

Steps S1a and S1b should reduce the dataset to focus primarily on geo-influencers whose followers are concentrated in a single time zone. Steps S2_t_1 and S3_t_2 help identify and correct instances where the baseline is making poor predictions that do not match using countries based on time and time+language features, respectively. If the top-ranked country from the baseline is within the set of countries it is returned as a prediction otherwise the second top country is used and so on.

There were 100,712 influencers with a self-reported location that could be resolved using GeoNames. The country that the influencer associates themselves with is used as ground truth. For example, *@BBCNews* has a self-reported location 'London' which is associated with GBR using GeoNames. Ordering by the most frequent country from followers' self-reported locations, the top five countries and associated percent of self-reported locations are USA: 28.15%, NGA: 10%, IND: 6.3%:, GBR: 5.92%, and KEN: 4.81%. Thus, USA is the top prediction used in baseline, but this does not match the set of possible countries from S2_t_1 and so NGA is used as it is the second best prediction; NGA has the same UTC as GBR and is thus within

the set of possible countries using time features. Language features from S3_t_2 can further constrain the prediction to the expected result, GBR.

In the above, we have described a procedure that results in a set of possible countries making up U_3. We also consider a point estimate prediction where a single country with the best cosine similarity is returned for a narrow UTC range $t_1 = 0.25$ (this point estimate is denoted as S3_Point in Table 6.2).

Table 6.2 shows how the baseline is constrained using each step of the pipeline. The second column shows precision across all influencers (100,712) and the fourth column shows precision across influencers not associated with the USA (37,908).

Table 6.2: Different Stages of Pipeline Improve Baseline Precision

	All P	**Count**	**Foreign P**	**Count**
Baseline	86.34	100708	65.71	37904
S1a	86.23	95500	65.61	36087
S1b	89.98	71643	76.97	28725
S2_1	90.23	70544	77.39	28018
S2_0.5	90.95	65708	78.98	25845
S2_0.25	92.23	60387	79.43	20753
S2_1+S3_0.95	91.49	64940	78.15	23315
S2_0.5+S3_0.95	92.78	57584	80.86	19721
S2_0.25+S3_0.95	94.54	50978	80.94	13395
S2_0.25+S3_Point	97.2	35518	86.59	6587

Table 6.2 illustrates that as time and language constraints are added the precision improves. Removing influencers that don't pass the sleep cycle test surprisingly doesn't help (row S1a), but getting rid of influencers with noisy PST predictions (row S1b) results in a big jump in precision.UTC offset information further constrains the baseline (rows S2_t_1). The highest precision is achieved when both UTC and language information are used.

6.5 Training Geocoder with Support for Foreign Languages

As already noted, the issue with the baseline Geocoder was that it handles only English based locations that have a match to GeoNames. Our proposed approach is to use the high confidence influencer predictions from the previous section as training data for an improved geocoder. This geocoder will automatically learn common ways persons refer to locations in their native tongue.

This new geocoder is based on a TF-IDF model, where the country is the document and the terms are the self-reported locations of the influencer's followers. It is possible to apply this model because if the influencer and their followers belong to the same country, then the followers' locations will capture common ways persons refer to locations within that country. TF-IDF vectors are generated using the Gensim package in Python[1].

Focusing on the top K TF-IDF features per country it is possible to verify that the model is generating reasonable vectors. Fig. 6.2 shows the top three features for several countries. These strings capture typical ways persons refer to locations within that country which can serve as useful features for geocoding type classifiers. The model also helps confirm that the countries, with which the influencer's followers were associated with, are indeed relevant.

Given a new influencer, the self-reported locations associated with the influencer's followers form a new document. A TF-IDF vector is built using self-reported location frequencies and the IDF component previously computed over the corpus of

[1]https://radimrehurek.com/gensim/

country	1	2	3
USA	charlottenc	chicago	louisvilleky
IND	india	newdelhiindia	mumbaiindia
PRT	portoportugal	portugal	lisboaportugal
ITA	italia	milano	bolognaemiliaromagna
JPN	日本	東京	日本東京
RUS	Москва	Россия	СанктПетербург
THA	ประเทศไทย	กรุงเทพมหานครประเทศไทย	bangkokthailand
TUR	İstanbul	istanbul	bodrum
ISR	israel	ישראל	telaviv
UKR	ukraine	Украина	kyiv
DEU	berlindeutschland	hamburg	berlingermany
KOR	republicofkorea	seoulrepublicofkorea	대한민국서울
MEX	monterreynuevoleón	méxico	mexico
KEN	nairobi	nairobikenya	kenya
ROU	romania	bucureştiromânia	bucharestromania
AFG	afghanistan	kabulafghanistan	kabul
GRC	attikigreece	thessalonikigreece	ΑττικήΕλλάς
PRI	puertorico	puertoricousa	sanjuanpuertorico
COL	barranquillacolombia	bogotádccolombia	calicolombia
ESP	madrid	madridcomunidaddemadrid	españa

Figure 6.2: Top three TF-IDF location features automatically learned from country documents. The benefit of this model is that it learns popular ways of referring to the country's locations in different languages and will include common phrases, abbreviations, and so on.

D documents. Cosine similarity is then used to return the country vector that is closest to the TF-IDF vector. Because the TF-IDF vectors may be very large, we recommend utilizing Latent Semantic Analysis (LSA) to reduce dimensionality to at most 500 terms (from literature 50-500 is recommended as a standard [89]). Fig. 6.3 highlights the overall approach.

In Table 6.3, the performance of the TF-IDF model is shown against the baseline from the previous section. TF-IDF model has a higher precision than original baseline with a clear jump in precision for foreign countries. This improved geocoder can be used to generate labels across additional influencers (that could not be predicted

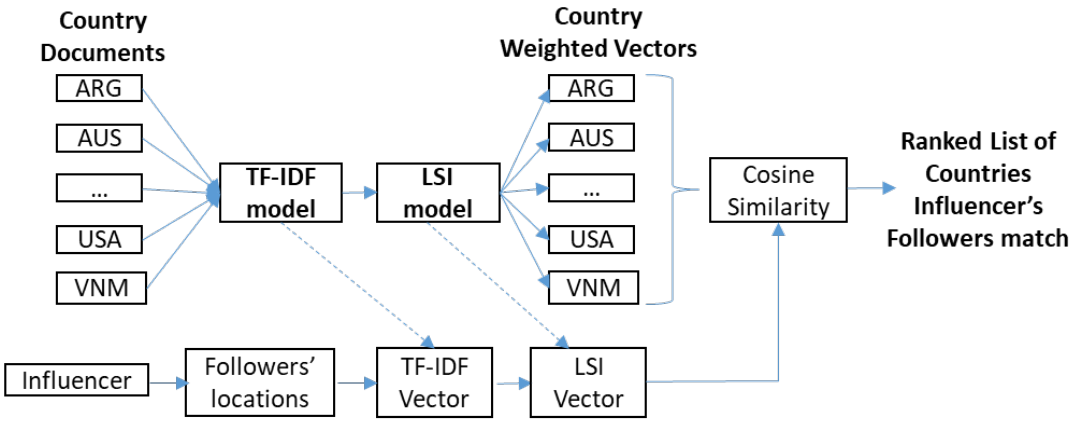


Figure 6.3: The process by which TF-IDF model is learned from geo-influencers associated with a country and used for predicting other geo-influencers.

using the original baseline). The additional labels can be verified using time and language features and the TF-IDF model can be further improved. The process can repeat until the process converges, that is, when the TF-IDF model no longer improves.

Table 6.3: TF-IDF model performance for Different Stages of Pipeline

	All P	**Count**	**Foreign P**	**Count**
Baseline	86.34	100708	65.71	37904
TF-IDF model	88.44	100711	81.75	37907

6.6 Conclusions

This chapter showed an application by which the geographic region can be labeled using only time and language features. The benefit of our approach is that these time and language features are universal and can be used across the whole Twitter-sphere. The labels have been used to train a high-level geocoder that has multilingual

support. The benefits are that the common ways that Twitter users report their locations are captured by this geocoder. The drawback to our method is that it works at a high level for predicting regions of the world at the country level or larger. We envision that the features proposed will be utilized for augmenting with other features as part of information fusion.

Chapter 7

Application 2: Repository of Influencers for Content Recommendation

7.1 Introduction

As discussed in previous chapters, a repository of location-aware influencers may be of interest for content recommendation and for studying location related communities from influencer's followers. This chapter describes the repository collected as part of this research, the features, and the visualization we employ. The repository consists of over three hundred thousand verified influencers that are tracked by *@verified*. For each user, the location information, time, and language features are recorded as was described in chapters 2, 5, and 6, respectively. Additionally, other demographics such as gender and race are considered.

7.2 Related Research

Popular variables associated with demographics are geographical location [5], age [90], gender [91], education [92, 93], income [94], ethnicity [95], and others. Typically, due to privacy concerns, these variables are not specifically stated. However, by fusing information with other sources, it is possible to characterize a group of

users with a certain level of confidence. For example from the US Census Bureau's Genealogy Project which publishes the frequency of popular surnames with their distribution per race/ethnicity, it is possible to characterize the percent of users that belong to a certain ethnic group [95]. A lot of the approaches deal with English-speakers [96].

Many additional variables may be inferred from online connections a user has that apply to any language. For example followers of @ESPN and @SportsCenter are more likely to be male, followers of @PlayStation are more likely to be kids, and followers of @ParentsMagazine are likely to have kids [97]. Political orientation may be discovered by whether the user is connected to known political representatives [98]. Timezone and language can be used for differentiating between countries. Our goal is to focus on those demographic features that can be used to characterize a large portion of the global population. Our repository focuses on self-reported location, gender, race, language, and time of the day their account was created.

7.3 Setting up the Repository

Implementation utilized Ubuntu 16.04 as the OS, Python 3.5 as the programming language, and MongoDB 3.6.5 as the NoSQL database. ElasticSearch and Kibana were used to search and visualize data of interest.

Fig. 7.1 illustrates the collection process for a single Twitter influencer. Each component from Fig. 7.1 is described below.

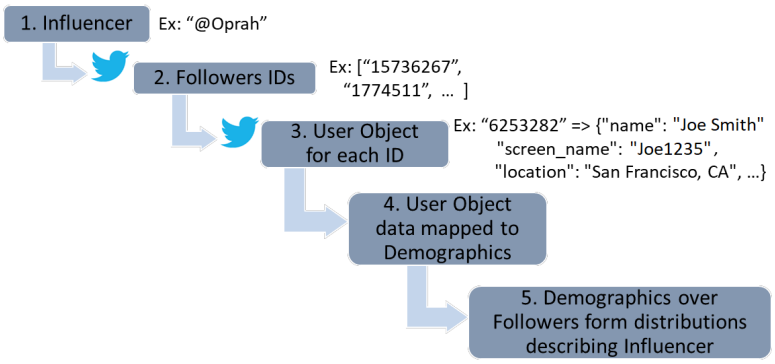


Figure 7.1: Collection Process.

- Influencer to Follower IDs: influencers are identified from @verified. For each influencer, up to 5000 followers are collected.

- IDs to User Objects: Fig 7.2 is a Snapshot of our database. Database stores all followers and all influencers collected in tables ‘userInfo’ and ‘followerInfo’ respectively. Example of User Object from userInfo table shown in Fig 7.3.

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size
followerInfo	373,116,407	621.1 B	215.8 GB	1	8.5 GB
followerInfoDemographics	373,116,407	309.0 B	107.4 GB	1	9.6 GB
userInfo	322,213	751.9 B	231.0 MB	1	7.3 MB
userInfoDemographics	322,213	343.7 B	105.6 MB	1	4.4 MB

Figure 7.2: Database holding Twitter User Objects for 322 thousand influencers and 373 million followers.

- User Object to Demographics: User’s textual location is matched against GeoNames city/country pairs, the self-reported name is matched against names with known gender and ethnicity. The associated demographic info, for all

```
_id: "50393960"
geo_enabled: false
verified: true
followers_count: 47177430
id_str: "50393960"
default_profile_image: false
listed_count: 122154
statuses_count: 3069
description: "Sharing things I'm learning through my foundation work and other inter..."
friends_count: 197
location: "Seattle, WA"
screenName: "BillGates"
lang: "en"
urlsInProfile: "http://www.gatesnotes.com/"
favourites_count: 116
name: "Bill Gates"
url: "https://t.co/AITfIziPWQ"
created_at: 2009-06-24 14:44:10.000
```


Figure 7.3: Example fields for Twitter User Object corresponding to @BillGates

followers and all influencers, is stored in tables 'userInfoDemographics' and 'followerInfoDemographics', respectively. If gender is present it is given by 'pctFemale' and 'pctMale'; race given by 'pctapi': Asian, 'pctblack': African American, 'pctwhite': Caucasian, 'pcthispanic': Hispanic.

- Demographics to Distributions: all influencer's followers with gender, race, location, language, time information are used to form corresponding frequency distributions. These distributions can then be used to understand the influencer's influence over the ordinary population by analyzing the percent of influencer's followers within certain gender, language, and so on.

ElasticSearch is utilized for loading and exploring data that is relevant to a specific scenario. Thus while MongoDB holds all of the data, ElasticSearch is used to search for a specific demographic. The end-user can utilize the Kibana visualization to form custom queries and zoom in and out on the map. Fig. 7.4 shows the visualization dashboard over followers for influencer @CNNEE (CNN Espanol) (similarly any one

of the 320K influencers can be visualized).

7.4 Features used in Repository

7.4.1 Location

Chapter 2 described an approach for geocoding a user's textual location. The rules of the classifier developed illustrated that it is important to consider whether a location that is matched contains both the city and state as this is less ambiguous than a city name by itself. Because Google's geocoder is limited, by the number of API calls it can freely make daily, we focused on matching locations that contain a known city/state or city/country for a high precision/low recall solution. For our task, it is better to focus on high precision locations given that verified influencers typically have thousands of followers which generally provides a large enough sample (as seen in section 4.7.2) to accurately pinpoint the influencer's city location.

GeoNames data is processed by (i) verifying each city entry to have a population above five thousand (806 entries were filtered out) and (ii) removing duplicate entries that refer to the same city by taking the most recent entry or one with the highest population (981 entries removed).

For each city, the City + Country Name, Country ISO, Country FIPS, and Country ISO3 are recorded as query strings. For example, for London UK these are the possible strings: 'londongbr', 'londonuk', 'londonunitedkingdom', 'londongb'. For the United States, we also search for City + State Name or State Abbreviation;

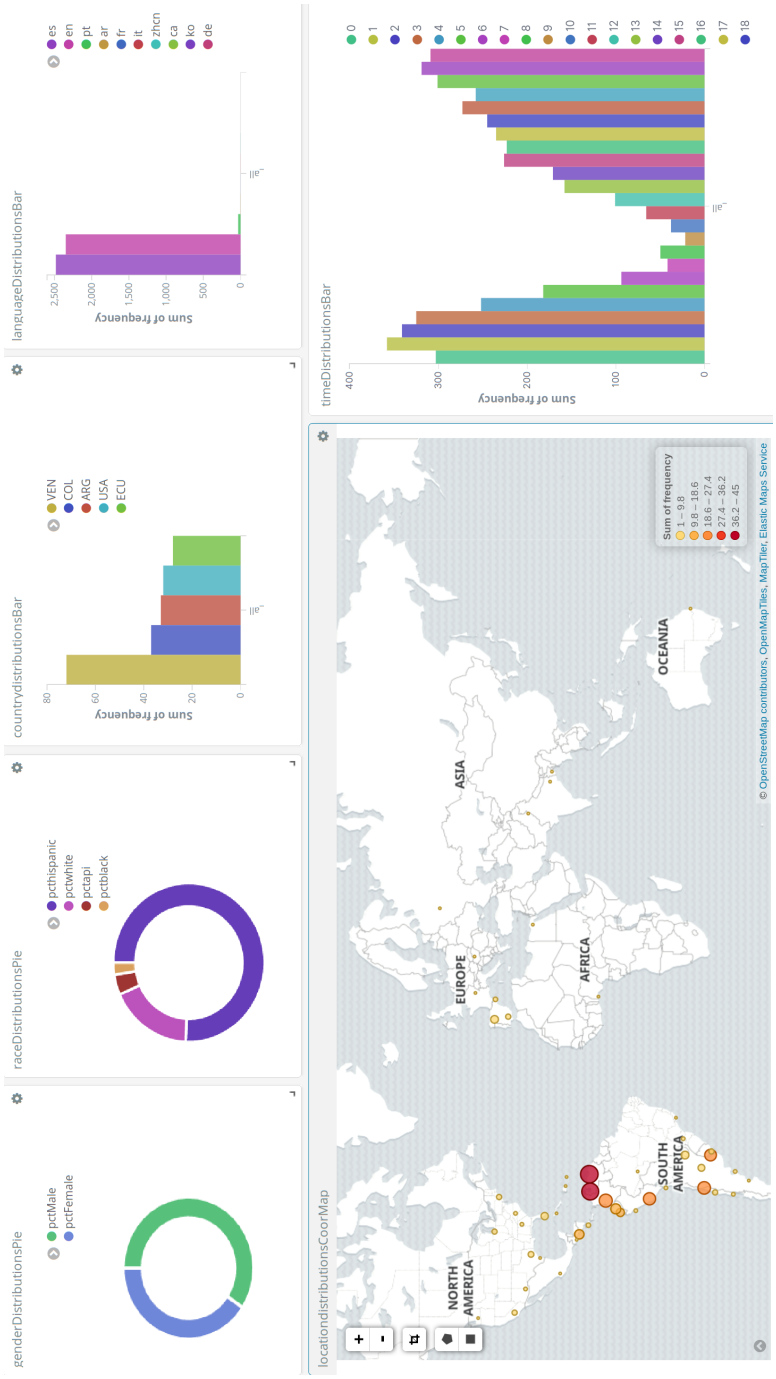


Figure 7.4: Visualizing influence by demographic. Understanding the influence of @CNNEE.

Table 7.1: Countries with most and least cities from GeoNames

Top 20 Country to City		**Bottom 20 Country to City**	
United States	7113	Saint Martin	1
India	3189	Sao Tome and Principe	1
Germany	2780	Guernsey	1
Russia	2525	Saint Kitts and Nevis	1
France	1972	Saint Barthelemy	1
Italy	1919	Saint Pierre and Miquelon	1
Brazil	1854	Jersey	1
Mexico	1725	Seychelles	1
United Kingdom	1603	Cook Islands	1
Spain	1302	Tonga	1
Philippines	1160	British Virgin Islands	1
China	842	Gibraltar	1
Australia	816	Palau	1
Romania	755	Grenada	1
Japan	739	Faroe Islands	1
Turkey	714	Antigua and Barbuda	1
Poland	661	Dominica	1
Ukraine	641	Barbados	1
Netherlands	484	Aland Islands	1
Colombia	484	Macao	1

for example ‘uticany’, ‘uticanewyork’. The country name is utilized for cities with a population over 100K: ‘syracuseunitedstates’, ‘syracuseus’, ‘syracuseusa’ (fips and iso equal in this example). This is done for cities where there is no other city with population over 100K (example ‘arlingtonus’ is not allowed since it can refer to Arlington TX or Arlington VA which both have a population over 100K). There are also cities such as New York City which have ‘City’ as part of the name, but that users may choose not to spell out; we allow city name variations with following tokens removed: ‘municipality’, ‘village’, ‘city’, ‘charter’, ‘township’, and ‘town’.

In all, there were 47119 unique cities and 156037 corresponding representations. Each follower’s self-reported location is turned to lowercase with punctuation and whitespace stripped out. Follower’s preprocessed location is utilized if it matches one of the 156037 corresponding representations. Table 7.1 shows the number of unique cities associated with each country.

In all 225 countries are represented. For each influencer, locations over all followers are used to form a location distribution. The cities in location distribution can be aggregated to generate a country distribution. Location and country distributions are used to identify influencers serving a specific region of the world.

7.4.2 Gender

Social Security Administration (SSA) provides popular female and male names[1]. The data contains the name, gender, and frequency. A specific name can be used

[1]https://www.ssa.gov/oact/babynames/names.zip

as a female and a male example from file: Emma, F, 19738 and Emma, M, 14. The frequency for gender divided by total frequency gives a percentage for how likely a particular name is to be male vs. female. Given frequencies for Emma, P(Emma, F) = 99.93% and P(Emma, M) = 0.071% as probabilities for female and male, respectively. Some names are on the borderline: Temiloluwa 55%, Carroll 45.5%, and Arley 57.3% (male probability).

This dataset consisted of 29910 names. For each Twitter follower, the first token of the name field is utilized. The token is converted to lowercase with punctuation stripped out. If it is contained within the names dataset then it is assigned a gender probability. For each influencer, the male and female probabilities over followers are added up and divided by the number of followers with name information.

7.4.3 Ethnicity

The Census Bureau identifies the last name to race mapping. The 2010 dataset provides Surnames Occurring 100 or more times with 162254 surnames[2]. We use the following four race categories: Non-Hispanic White Alone (White), Non-Hispanic Black or African American Alone (Black), Non-Hispanic Asian and Native Hawaiian and Other Pacific Islander Alone (Asian) and Hispanic or Latino origin (Hispanic) (there are two more categories, but those have too few data points, see reference [99]). For each Twitter follower, if the self-reported name contains two tokens then the second token of the name field is utilized. The token is converted to lowercase with punctuation stripped out. If it is contained within the surnames dataset then

[2]https://www2.census.gov/topics/genealogy/2010surnames/names.zip

Table 7.2: Top 20 Languages Across the Dataset

language code	follower count	overall percent
en	244313028	64.7
es	33680216	8.92
ja	13899866	3.68
ar	11024275	2.92
pt	10550007	2.79
fr	10374538	2.75
tr	9024648	2.39
ru	7717812	2.04
engb	5279311	1.4
none	4495906	1.19
id	4373498	1.16
de	4154066	1.1
it	3094860	0.82
zhcn	2942235	0.78
nl	1965084	0.52
ko	1340514	0.35
vi	1154730	0.31
pl	959457	0.25
th	933313	0.25
sv	753957	0.2
zhtw	613812	0.16

it is utilized. For each influencer, the ethnicity probabilities over followers are added up and divided by the number of followers with ethnicity information.

7.4.4 Time and Language

The time distribution over influencer's followers is generated as described in Section 5.2. The language field is available for 98.81% of users. Table 7.2 shows the top 20 languages over 373116407 users. The table shows that about 64.7% of users prefer

English, 8.92% prefer Spanish, 3.68% prefer Japanese, and so on. For each influencer, the ratio of followers preferring a specific language is recorded. Influencers who target a specific country such as Spain are expected to have an above average number of Spanish speakers.

7.4.5 DBPedia

DBPedia is a large publicly available resource that is used for leveraging external data related to Twitter users. We are interested in those DBPedia pages that have a matching Twitter screenname. Sometimes this will be part of infobox data, other times this data can be predicted. Reference [100] is an example of a recent paper that attempts to match DBPedia pages to Twitter screennames. For training and test data, they try the top-ranked Twitter profile (if any) returned by Twitter when queried with the DBPedia entity name. 893,446 DBPedia entities were matched to 630,767 Twitter candidates and a Deep Neural Net (DNN) used to align 169,748 of these. For evaluation, their gold standard is made of those DBpedia pages where the Twitter screenname is explicitly stated consisting of 56,133 alignments from English DBpedia entities (40,967 persons, 15,166 organizations).

We utilize SocialLink latest Gold and latest predicted mappings that had a score of 0.75 or greater (75% or higher confidence). Out of these influencers, 44450 appear in our dataset (12,771 from gold and 36,029 from predicted, some influencers appear in both lists).

DBPedia uses a Virtuoso RDF triple store that requires SPARQL queries. If a

Table 7.3: DBPedia Categories of Interest

DBPedia ID	label	Number of Values
P27	country of citizenship	224
P569	date of birth	9245
P172	ethnic group	213
P1412	languages spoken, written or signed	141
P21	sex or gender	8
P19	place of birth	9245
P140	religion	124
P69	educated at	5660
P735	given name	6222
P734	family name	10420
P106	occupation	1207
P641	sport	194
P54	member of sports team	5143
P136	genre	649

page exists on Wikipedia such as: 'https://en.wikipedia.org/wiki/Oprah_Winfrey' it is also available on DBPedia 'http://dbpedia.org/page/Oprah_Winfrey' (unique id being 'Oprah_Winfrey'). A Python client library called Wikidata[3] was used.

Each DBPedia page is represented by Q#. Each result contains the label, description, claims, and references. The first step was to collect all claims for each DBPedia page that had a link to a Twitter influencer. The frequency of all unique claims was recorded with some of the items of interest from Table 7.3.

Each item listed points to a separate DBPedia page. For example, there are is a separate page for each country, where each page provides additional information such as the country's coordinates, population, inception date, and other information. In this way, for each influencer, we attempt to collect all categories shown in Table

[3]https://github.com/dahlia/wikidata

Table 7.4: For each Feature, the Average and Standard Deviation are computed across 320K influencers. This sets a threshold for influencers that capture above the average demographic. Number of influencers recommended for each demographic feature shown in last column.

Feature	**AVG**	**STD**	**T = AVG+STD**	**Influencers**
Gender Male	61.69%	16.37%	78.06%	61988
Gender Female	38.34%	16.35%	54.69%	46935
Ethnicity White	60.39%	17.89%	78.28%	19179
Ethnicity Black	10.95%	5.09%	16.04%	26256
Ethnicity Hispanic	13.65%	16.26%	29.92%	33010
Ethnicity Asian	11.72%	15.03%	26.75%	33300

7.3 and then for each category we collect additional information that characterizes the category. This leads to a rich set of additional data available for influencers that are popular enough to be described on Wikipedia.

7.5 Example Rankings by Demographic Group

This section illustrates how the repository can be used for content recommendation for a certain demographic; where the demographic is based on gender, ethnicity, language, location, or a combination of these.

For each demographic feature, our approach is to (i) for each influencer to record the percent of followers that fall into the demographic, (ii) record the average and standard deviation of the percent across all influencers, and (iii) set threshold equal to average plus one standard deviation. Table 7.4 shows the computed thresholds for demographic features related to gender and ethnicity.

Table 7.5: Top 10 Most Popular Influencers for each Gender.

Top 10 Influencers with over 78.06% Followers being male

screenName	followers	name	%Male
narendramodi	47235007	Chowkidar Narendra Modi	81.39
SrBachchan	37091473	Amitabh Bachchan	80.41
BeingSalmanKhan	36816795	Salman Khan	81.18
SportsCenter	35437313	SportsCenter	78.58
realmadrid	31984155	Real Madrid C.F.	85.34
akshaykumar	30511248	Akshay Kumar	78.97
FCBarcelona	29627195	FC Barcelona	85.0
imVkohli	29436203	Virat Kohli	83.95
sachin_rt	29057313	Sachin Tendulkar	84.32
PMOIndia	28895265	PMO India	83.17

Top 10 Influencers with over 54.69% Followers being female

screenName	followers	name	%Female
justinbieber	105481835	Justin Bieber	64.03
TheEllenShow	77630706	Ellen DeGeneres	62.09
ArianaGrande	62676030	Ariana Grande	59.2
KimKardashian	60662411	Kim Kardashian West	57.68
selenagomez	57579185	Selena Gomez	58.51
jimmyfallon	51109039	jimmy fallon	55.94
MileyCyrus	42505219	Miley Ray Cyrus	58.68
NiallOfficial	39292796	Niall Horan	62.67
Harry_Styles	33315825	Harry Styles.	79.25
Louis_Tomlinson	33232768	Louis Tomlinson	67.75

Gender– The gender was computed for 185078761 out of 373116407 (49.6%) of users. Across all users 57.6% were male. Across all influencers on average 61.69% of followers were male with a standard deviation of 16.37% (threshold = 61.69%+16.37% = 78.07%). The last column shows that there were 61988 influencers whose audience is over 78.07% male. Top ten influencers exceeding this threshold and ordered by the number of followers shown in Table 7.5 (top). Similarly, 46935 influencers whose audience is over 54.69% female, with the corresponding top ten influencers shown in Table 7.5 (bottom).

Ethnicity– For ethnicity, 125780191 out of 373116407 (33.71%) users had, as a second token in their name, a surname that maps to a known ethnicity. Of these 56.54% were White, 18.21% Hispanic, 13.98% Asian, and 10.27% Black. Table 7.4 shows the computed thresholds for each ethnicity across influencers' followers. Using these thresholds the top influencers for each ethnicity shown in Table 7.6.

Location– There are over 200 countries. For each influencer, the country distribution is formed from self-reported locations of the followers that mention city and country. There are a total of 36980202 out of 373116407 (9.91%) followers listing such well-formed locations. The known countries are used to generate a distribution that can be used to focus on specific influencers. As an example, Table 7.7 (top) shows the top ten influencers associated with Indonesia (in repository 1663 influencers had over 50.0% of their followers from Indonesia).

Language– Language information is available for over 98.8% of followers and is thus a powerful feature since there is a large sample size of followers with the field. Table 7.7 (bottom) shows the top ten influencers whose followers are 50% or more Spanish.

Table 7.6: Most Popular Influencers for each Ethnicity.

Top 6 Influencers with over 78.28% Followers being Caucasian

screenName	followers	name	%Caucasian
lorenzojova	3819516	Lorenzo Jovanotti	79.13
repubblica	2857050	la Repubblica	78.6
SPIEGELONLINE	2529707	SPIEGEL ONLINE	78.76
beppe_grillo	2470978	Beppe Grillo	79.78
MarroneEmma	2469369	Emma Marrone	80.44
radiodeejay	2278769	Radio Deejay	81.63

Top 6 Influencers with over 16.04% Followers being African American

screenName	followers	name	%Black
Oprah	42164712	Oprah Winfrey	16.51
KevinHart4real	35194359	Kevin Hart	16.12
LilTunechi	34235040	Lil Wayne WEEZY F	19.82
wizkhalifa	34028999	Wiz Khalifa	16.51
chrisbrown	30245761	Chris Brown	19.24
aliciakeys	30023640	Alicia Keys	18.01

Top 6 Influencers with over 29.92% Followers being Hispanic

screenName	followers	name	%Hispanic
shakira	51127526	Shakira	35.82
Louis_Tomlinson	33232768	Louis Tomlinson	33.29
realmadrid	31984155	Real Madrid C.F.	31.11
pitbull	26128582	Pitbull	31.71
ricky_martin	20383994	Ricky Martin	56.29
AlejandroSanz	19516691	Alejandro Sanz	71.78

Top 6 Influencers with over 26.75% Followers being Asian

screenName	followers	name	%Asian
narendramodi	47235007	Chowkidar Narendra Modi	78.17
BillGates	47177430	Bill Gates	33.24
iamsrk	38104866	Shah Rukh Khan	73.74
SrBachchan	37091473	Amitabh Bachchan	76.75
BeingSalmanKhan	36816795	Salman Khan	72.79
akshaykumar	30511248	Akshay Kumar	73.73

Table 7.7: Top 10 Influencers from Indonesia (top), speaking Spanish (bottom)

Top 10 Influencers with over 50% Followers from Indonesia

screenName	followers	profile location	%IND
agnezmo	17649171	Los Angeles	91.41
radityadika	15734925	Jakarta Selatan, DKI Jakarta	97.6
detikcom	15102845	Jakarta, Indonesia	95.31
LunaMaya26	11651186	INDONESIA	94.23
cinema21	11444672	Jakarta, Indonesia	94.94
jokowi	11384071	Jakarta	93.13
sherinasinna	10781243		92.03
Metro_TV	10395864	UT:-6.186977, 106.759125	93.43
SBYudhoyono	10086130		95.0
afgansyah_reza	9926114	Indonesia	94.47

Top 10 Influencers with over 50% Spanish Followers

screenName	followers	profile location	%Spanish
CNNEE	17232226	En todas partes	50.74
TwitterLatAm	14948461	América Latina	54.1
juanes	11599148		52.2
PaulinaRubio	11209842		50.29
CHAYANNEMUSIC	9290096	Miami, Florida	54.63
SofiaVergara	8990538		51.63
muyinteresante	8368847	Spain	65.47
werevertumorro	8330870	México, DF	58.89
AristeguiOnline	8275955	México, DF	60.07
CarlosLoret	8033311	México, DF.	56.09

The location and description fields of these influencers match Latin America, Mexico, Spain - locations with mostly Spanish speaking audiences. There were 22067 influencers with over 50.0% of their followers preferring Spanish.

Table 7.8: Verified vs. Unverified number of messages (top) vs. followers (bottom)

Verified vs. Unverified Users number of messages posted

Range	**Pct**	**Unverified (avg, std)**	**Pct**	**Verified (avg, std)**
[0,0]	24.627	(0.0, 0.0)	0.143	(0.0, 0.0)
[0,10]	52.125	(1.88, 2.63)	0.668	(3.58, 3.28)
[10,100]	20.663	(36.73, 24.57)	2.466	(52.19, 26.43)
[100,1000]	15.390	(370.92, 243.46)	14.997	(500.69, 260.57)
[1000,10000]	9.999	(3412.65, 2293.9)	47.675	(4365.28, 2518.2)
[10000,100000]	2.793	(24997.56, 17105.79)	31.976	(27613.0, 18946.36)
[100000,1000000]	0.094	(165696.91, 88231.33)	2.257	(193657.95, 124088.35)
[1000000,Inf]	0.0002	(1481857.94, 1334891.99)	0.0394	(2604262.9, 4528258.8)
[0, Inf]	100	(1264.22, 8248.86)	100	(16381.72, 109704.1)

Verified vs. Unverified Users number of followers

Range	**Pct**	**Unverified (avg, std)**	**Pct**	**Verified (avg, std)**
[0,0]	15.815	(0.0, 0.0)	0.0012	(0.0, 0.0)
[0,10]	55.163	(2.72, 2.86)	0.0722	(8.05, 2.67)
[10,100]	29.49	(34.9, 23.94)	0.729	(42.23, 27.67)
[100,1000]	14.903	(305.88, 205.6)	7.3415	(587.87, 250.8)
[1000,10000]	1.928	(2409.18, 1767.89)	41.253	(4361.78, 2504.94)
[10000,100000]	0.158	(24420.97, 17754.61)	36.922	(33291.61, 22652.99)
[100000,1000000]	0.0105	(218116.84, 151487.27)	11.663	(294429.2, 208785.86)
[1000000,Inf]	0.0003	(1862548.7, 1183416.8)	2.057	(3374510.0, 5580356.3)
[0, Inf]	100	(170.97, 4921.16)	100	(117889.54, 936335.99)

7.6 Verified vs. Unverified User Comparison

In this section, the differences between verified (over 322 thousand users) vs. unverified (over 373 million users) are analyzed. The average number of messages, friends, and followers that a user has are analyzed. In previous chapters, it was claimed that most Twitter users are silent consumers of information. Table 7.8 (top) shows the percent of users that are posting a specific rate defined by the range (min, max) for unverified vs. verified users. From Table about 1/4 (24.63%) never posted any content, about 1/3 (33.01%) posted less than 1 message, 1/2 (52.12%) posted 10 or less, and so on. On average, the verified users post 16381 vs. 1264 messages for unverified users.

In a similar fashion, the average number of followers for verified vs. unverified users is shown in Table 7.8 (bottom). In section 4.7.2 it was shown that at least 500 followers are needed to get a large enough sample for computing the central geographic location that the influencer serves. Using at least 500 influencers 3.581% of verified vs. 97.077% of unverified users are lost (thus a vast majority of verified users pass this threshold). The table shows that verified users have a lot more followers on average 117889 vs. 171 for unverified users.

Fig. 7.5 is a depiction using ranges from Table 7.8 differentiating verified vs. unverified over (i) the total number of messages (top chart), (ii) the total number of followers (middle chart), and (iii) the total number of friends (bottom chart). Fig. 7.5 bottom shows that a good portion of verified users is also engaged in following others (forming friend connections).

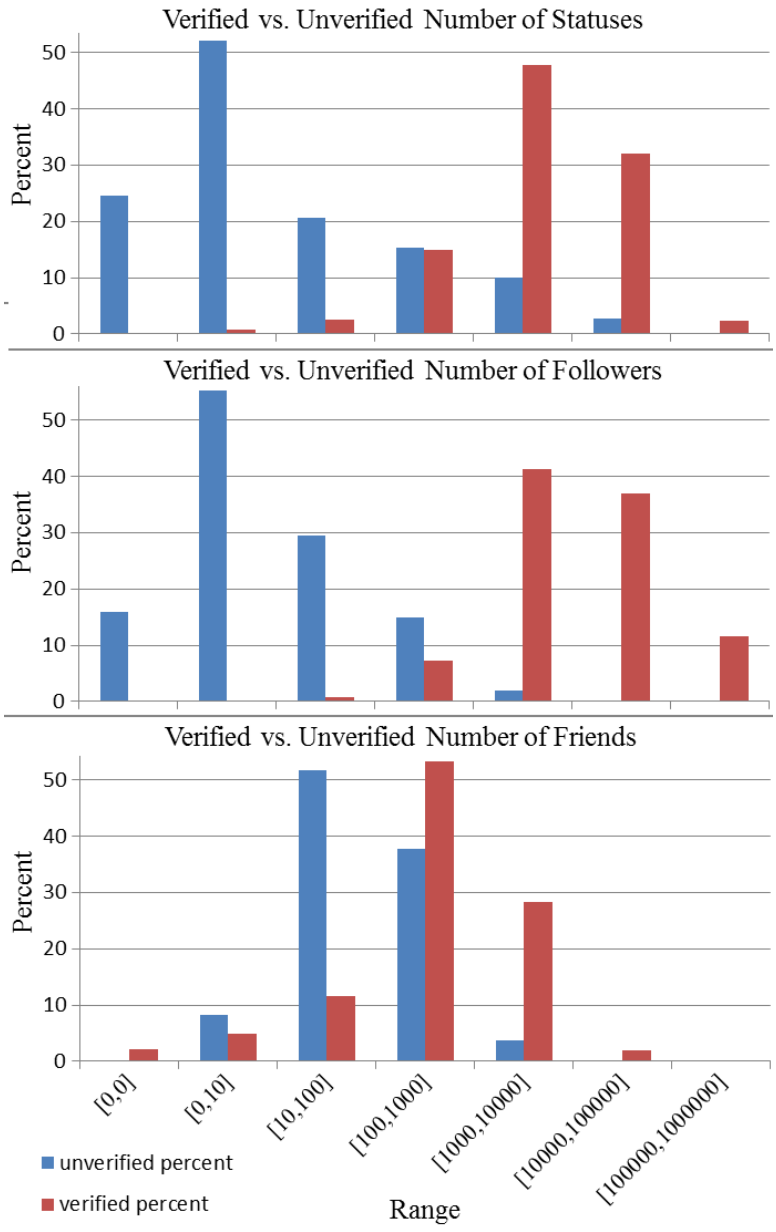


Figure 7.5: Comparison of Verified vs. Unverified users using: (top) number of messages posted, (middle) total followers, and (bottom) total friends.

In summary, verified influencers are more likely to have more messages, more followers, and more friends. Given that most users are passive and do not generate much message traffic supports our argument for focusing on follower-followee link structure and profile metadata for mapping influence.

7.7 Conclusions

The social media site that is Twitter is a big data challenge. Twitter generates 500 million messages on a daily basis. It can be a daunting task to figure out how to collect the information that corresponds to the specific demographic of interest. This chapter presented a tool for quickly identifying influencers serving a specific demographic. This is important for content recommendation where we can identify influencers based on the composition of their audience using gender, ethnicity, language, location, or a combination of these. Also, by focusing on the followers of these influencers a community that is representative of the population of interest can be quickly established.

Chapter 8

Conclusions

The main theme of our research is in understanding the geographic spread of a group; where the group can be made up of Twitter users or messages. We have explored (i) location - improving geocoding by customizing an existing geocoder as well as training a high-level geocoder with multilingual support, (ii) time - predicting whether a group is local or global and the associated UTC offset, (iii) inferring location - leverage Twitter network connections to known geo-influencers (users that cater their content to a specific geographic area).

Google search was leveraged for identifying an initial set of geo-influencers related to a city of interest; it was shown that the geo-influencers' followers are representative of the city's users (i.e. most of the followers are physically located in the city). Location features are necessary for city-level geocoding, but time and language can also help reason about the geographic spread. Research showed that using only time-based features is enough to associate top trending topics and persons with different geographical regions. The new time-based features are not just limited to inferring location, but can also be used for inferring link creation times and for studying the evolution of influencer's popularity.

By maintaining a repository of geo-influencers it is possible to quickly leverage, as a starting point, those influencers within the geographic location of interest vs.

having to discover them in a time-intensive collection from scratch (this targetted collection is 100x faster). It was also shown how the repository can be used for filtering influencers based on their audience's demographics related to location, language, gender, and ethnicity. This research is important because it enables multiple applications, each of which were shown in our research: (i) content recommendation, (ii) community detection, (iii) inferring the location of users based on the link structure to geo-influencers, and (iv) studying the evolution of influencer's popularity. Main portions of the code are hosted on GitHub[1].

[1]https://github.com/apanasyu

Bibliography

[1] Karami, Amir, et al. "Twitter and research: a systematic literature review through text mining." IEEE Access 8 (2020): 67698-67717.

[2] Antonakaki, Despoina, Paraskevi Fragopoulou, and Sotiris Ioannidis. "A survey of Twitter research: Data model, graph structure, sentiment analysis and attacks." Expert Systems with Applications 164 (2021): 114006.

[3] Zheng, Xin, Jialong Han, and Aixin Sun. "A survey of location prediction on Twitter." IEEE Transactions on Knowledge and Data Engineering (2018).

[4] Graham, Mark, Scott A. Hale, and Devin Gaffney. "Where in the world are you? Geolocation and language identification in Twitter." The Professional Geographer 66.4 (2014): 568-578.

[5] Mislove, Alan, et al. "Understanding the Demographics of Twitter Users." ICWSM 11.5th (2011): 25.

[6] Stephens, Monica, and Ate Poorthuis. "Follow thy neighbor: Connecting the social and the spatial networks on Twitter." Computers, Environment and Urban Systems 53 (2015): 87-95.

[7] Beauchamp, Nicholas. "Predicting and interpolating state-level polls using Twitter textual data." American Journal of Political Science 61.2 (2017): 490-503.

[8] Byrd, Kenny, et al. "Mining Twitter data for influenza detection and surveillance." Proceedings of the International Workshop on Software Engineering in Healthcare Systems. ACM, 2016.

[9] Zubiaga, Arkaitz, et al. "Towards real-time, country-level location classification of worldwide tweets." IEEE Transactions on Knowledge and Data Engineering 29.9 (2017): 2053-2066.

[10] Jurgens, David, et al. "Geolocation Prediction in Twitter Using Social Networks: A Critical Analysis and Review of Current Practice." ICWSM 15 (2015): 188-197.

[11] Wong, Charlene A., et al. "Twitter sentiment predicts Affordable Care Act marketplace enrollment." Journal of medical Internet research 17.2 (2015).

[12] Matci, Dilek Küçük, and Uğur Avdan. "Address standardization using the natural language process for improving geocoding results." Computers, Environment and Urban Systems (2018).

[13] Jurgens, David. "That's What Friends Are For: Inferring Location in Online Social Media Platforms Based on Social Relationships." ICWSM 13.13 (2013): 273-282.

[14] Malik, Momin M., et al. "Population bias in geotagged tweets." People 1.3,759.710 (2015): 3-759.

[15] Tasse, Dan, et al. "State of the Geotags: Motivations and Recent Changes." ICWSM. 2017.

[16] Prasetyo, Philips., et al. "On analyzing geotagged tweets for location-based patterns." Proceedings of the 17th International Conference on Distributed Computing and Networking. ACM, 2016.

[17] Laniado, David, et al. "The Impact of Geographic Distance on Online Social Interactions." Information Systems Frontiers(2017): 1-16.

[18] Singh, Sushant K. "Evaluating two freely available geocoding tools for geographical inconsistencies and geocoding errors." Open Geospatial Data, Software and Standards 2.1 (2017): 11

[19] Vincenty, Thaddeus. "Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations." Survey review 23.176 (1975): 88-93.

[20] Miyazaki, Taro, et al. "Twitter Geolocation using Knowledge-Based Methods." Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text. 2018.

[21] Riquelme, Fabián, and Pablo González-Cantergiani. "Measuring user influence on Twitter: A survey." Information Processing & Management 52.5 (2016): 949-975.

[22] Bouros, Panagiotis, Dimitris Sacharidis, and Nikos Bikakis. "Regionally influential users in location-aware social networks." Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. ACM, 2014.

[23] Li, Guoliang, et al. "Efficient location-aware influence maximization." Proceedings of the 2014 ACM SIGMOD international conference on Management of data. ACM, 2014.

[24] Darmon, David, Elisa Omodei, and Joshua Garland. “Followers are not enough: A multifaceted approach to community detection in online social networks.“ PloS one 10.8 (2015): e0134860.

[25] Java A, Song X, Finin T, Tseng B (2009) Why we twitter: an analysis of a microblogging community. In: Advances in Web Mining and Web Usage Analysis, Springer. pp. 118–138.

[26] Wang, T.-C.; Phoa, F.K.H.: A scanning method for detecting clustering pattern of both attribute and structure in social networks. Phys. A Stat. Mech. Appl. 445, 295–309 (2016)

[27] Yang, J.; McAuley, J.; Leskovec, J.: Community detection in networks with node attributes. In: IEEE 13th International Conference on Data mining (ICDM), pp. 1151–1156. IEEE (2013)

[28] Zhang, F.; Li, J.; Li, F.; Xu, M.; Xu, R.; He, X.: Community detection based on links and node features in social networks. In: He, X., Luo, S., Tao, D., Xu, C., Yang, J., Hasan, M.A. (eds.) MultiMedia Modeling. MMM 2015. Lecture Notes in Computer Science, vol. 8935. Springer, Cham (2015)

[29] Mucha, Peter J., et al. “Community structure in time-dependent, multiscale, and multiplex networks.“ science 328.5980 (2010): 876-878.

[30] Hossain, Nabil, et al. “Inferring fine-grained details on user activities and home location from social media: Detecting drinking-while-tweeting patterns in communities.“ arXiv preprint arXiv:1603.03181 (2016).

[31] Gurajala, Supraja, et al. “Fake Twitter accounts: profile characteristics obtained using an activity-based pattern detection approach.“ Proceedings of the 2015 International Conference on Social Media & Society. ACM, 2015.

[32] Bakillah, Mohamed, Ren-Yu Li, and Steve HL Liang. “Geo-located community detection in Twitter with enhanced fast-greedy optimization of modularity: the case study of typhoon Haiyan.“ International Journal of Geographical Information Science 29.2 (2015): 258-279.

[33] Maleewong, Krissada. “An analysis of influential users for predicting the popularity of news tweets.“ Pacific Rim International Conference on Artificial Intelligence. Springer, Cham, 2016.

[34] Xiao, Feng, Tomoya Noro, and Takehiro Tokuda. “Finding News-Topic Oriented Influential Twitter Users Based on Topic Related Hashtag Community Detection.“ J. Web Eng. 13.5&6 (2014): 405-429.

[35] Cha, Meeyoung, et al. “Measuring user influence in twitter: The million follower fallacy.“ Icwsm 10.10-17 (2010): 30.

[36] Jabeur, Lamjed Ben, Lynda Tamine, and Mohand Boughanem. “Active microbloggers: Identifying influencers, leaders and discussers in microblogging networks.“ International Symposium on String Processing and Information Retrieval. Springer, Berlin, Heidelberg, 2012.

[37] Rout, Dominic, et al. “Where’s@ wally?: a classification approach to geolocating users based on their social ties.“ Proceedings of the 24th ACM Conference on Hypertext and Social Media. ACM, 2013.

[38] Freeman, Linton C. “Centrality in social networks conceptual clarification.“ Social networks 1.3 (1978): 215-239.

[39] Gayo-Avello, Daniel. “Nepotistic relationships in twitter and their impact on rank prestige algorithms.“ Information Processing & Management 49.6 (2013): 1250-1280.

[40] Weng, Jianshu, et al. “Twitterrank: finding topic-sensitive influential twitterers.“ Proceedings of the third ACM international conference on Web search and data mining. ACM, 2010.

[41] Silva, Arlei, et al. “ProfileRank: finding relevant content and influential users based on information diffusion.“ Proceedings of the 7th Workshop on Social Network Mining and Analysis. ACM, 2013.

[42] Wei, Hong, Jagan Sankaranarayanan, and Hanan Samet. “Measuring Spatial Influence of Twitter Users by Interactions.“ Proceedings of the 1st ACM SIGSPATIAL Workshop on Analytics for Local Events and News. ACM, 2017.

[43] Subrahmanian, V. S., et al. “The DARPA Twitter bot challenge.“ Computer 49.6 (2016): 38-46.

[44] Li, Yuchen, et al. “Influence maximization on social graphs: A survey.“ IEEE Transactions on Knowledge and Data Engineering (2018).

[45] Su, Sen, et al. “Location-aware targeted influence maximization in social networks.“ Journal of the Association for Information Science and Technology 69.2 (2018): 229-241.

[46] Panasyuk, Aleksey, Edmund Yu, and Kishan Mehrotra. “Augmenting Google Search in Ranking Twitter Users.“ Semantic Computing (ICSC), 2019 IEEE 13th International Conference on. IEEE, 2019.

[47] Li, Yuchen, Dongxiang Zhang, and Kian-Lee Tan. “Real-time targeted influence maximization for online advertisements.“ Proceedings of the VLDB Endowment 8.10 (2015): 1070-1081.

[48] Seabrook, Elizabeth M., Margaret L. Kern, and Nikki S. Rickard. “Social networking sites, depression, and anxiety: a systematic review.“ JMIR mental health 3.4 (2016).

[49] Aghababaei, Somayyeh, and Masoud Makrehchi. “Mining social media content for crime prediction.“ 2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI). IEEE, 2016.

[50] Guo, Weisi, et al. “Understanding happiness in cities using Twitter: Jobs, children, and transport.“ 2016 IEEE International Smart Cities Conference (ISC2). IEEE, 2016.

[51] Panasyuk, Aleksey, Edmund Yu, and Kishan Mehrotra. “Improving Geocoding for City-level Locations.“ Semantic Computing (ICSC), 2019 IEEE 13th International Conference on. IEEE, 2019.

[52] Compton, Ryan, David Jurgens, and David Allen. “Geotagging one hundred million twitter accounts with total variation minimization.“ Big Data (Big Data), 2014 IEEE International Conference on. IEEE, 2014.

[53] Kariryaa, Ankit, et al. “Defining and Predicting the Localness of Volunteered Geographic Information using Ground Truth Data.“ Proc. CHI. Vol. 18. 2018.

[54] Leetaru, Kalev, et al. “Mapping the global Twitter heartbeat: The geography of Twitter.“ First Monday 18.5 (2013).

[55] Ferrara, Emilio, et al. “The rise of social bots.“ Communications of the ACM 59.7 (2016): 96-104.

[56] Webber, William, Alistair Moffat, and Justin Zobel. “A similarity measure for indefinite rankings.“ ACM Transactions on Information Systems (TOIS) 28.4 (2010): 20.)

[57] B. Han and A. Srinivasan, ’Your friends have more friends than you do: Identifying influential mobile users through random-walk sampling,’ in Proc. 13th MOBIHOC, 2012, pp. 5–14.

[58] Wang, Yufeng, et al. “PPRank: economically selecting initial users for influence maximization in social networks.“ IEEE Systems Journal 11.4 (2017): 2279-2290.

[59] Li, Xiao, et al. “Community-based seeds selection algorithm for location aware influence maximization.“ Neurocomputing 275 (2018): 1601-1613.

[60] Çıtlak, Oğuzhan, Murat Dörterler, and İbrahim Alper Doğru. ”A survey on detecting spam accounts on Twitter network.” Social Network Analysis and Mining 9.1 (2019): 1-13.

[61] Wu, Tingmin, et al. ”Twitter spam detection: Survey of new approaches and comparative study.” Computers & Security 76 (2018): 265-284.

[62] Yang, Kai-Cheng, et al. ”Scalable and generalizable social bot detection through data selection.” Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 34. No. 01. 2020.

[63] Craswell, Nick, Arjen P. De Vries, and Ian Soboroff. “Overview of the TREC 2005 Enterprise Track.“ Trec. Vol. 5. 2005.

[64] Husain, Omayma, et al. “Expert Finding Systems: A Systematic Review.“ Applied Sciences 9.20 (2019): 4250.

[65] T. Lappas, K. Liu, and E. Terzi. A survey of algorithms and systems for expert location in social networks. In Social Network Data Analytics. Springer, 2011.

[66] Page, Lawrence, et al. The PageRank citation ranking: Bringing order to the web. Stanford InfoLab, 1999.

[67] Kleinberg, Jon M. “Authoritative sources in a hyperlinked environment.“ Journal of the ACM (JACM) 46.5 (1999): 604-632.

[68] Romero, Daniel M., et al. “Influence and passivity in social media.“ Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, Berlin, Heidelberg, 2011.

[69] Pal, Aditya, and Scott Counts. “Identifying topical authorities in microblogs.“ Proceedings of the fourth ACM international conference on Web search and data mining. 2011.

[70] Ghosh, Saptarshi, et al. “Cognos: crowdsourcing search for topic experts in microblogs.“ Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval. 2012.

[71] Cheng, Zhiyuan, et al. “Who is the barbecue king of texas? A geo-spatial approach to finding local experts on twitter.“ Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval. 2014.

[72] Mourad, Ahmed, et al. “A practical guide for the effective evaluation of twitter user geolocation.“ ACM Transactions on Social Computing 2.3 (2019): 1-23.

[73] Li, Wen, Carsten Eickhoff, and Arjen P. de Vries. “Geo-spatial domain expertise in microblogs.“ European Conference on Information Retrieval. Springer, Cham, 2014.

[74] Li, Wen, Carsten Eickhoff, and Arjen P. de Vries. “Probabilistic local expert retrieval.“ European Conference on Information Retrieval. Springer, Cham, 2016.

[75] Niu, Wei, Zhijiao Liu, and James Caverlee. “On local expert discovery via geo-located crowds, queries, and candidates.“ ACM Transactions on Spatial Algorithms and Systems (TSAS) 2.4 (2016): 1-24.

[76] Yochum, Phatpicha, et al. ”Linked open data in location-based recommendation system on tourism domain: A survey.” IEEE Access 8 (2020): 16409-16439.

[77] Luceri, Luca, et al. ”Measurement and control of geo-location privacy on Twitter.” Online social networks and media 17 (2020): 100078.

[78] Zola, Paola, Costantino Ragno, and Paulo Cortez. ”A Google Trends spatial clustering approach for a worldwide Twitter user geolocation.” Information Processing & Management 57.6 (2020): 102312.

[79] Singh, Jyoti Prakash, et al. ”Event classification and location prediction from tweets during disasters.” Annals of Operations Research 283.1 (2019): 737-757.

[80] Paule, Jorge David Gonzalez, Yeran Sun, and Yashar Moshfeghi. ”On fine-grained geolocalisation of tweets and real-time traffic incident detection.” Information Processing & Management 56.3 (2019): 1119-1132.

[81] Inkpen, Diana, et al. “Location detection and disambiguation from twitter messages.“ Journal of Intelligent Information Systems 49.2 (2017): 237-253.

[82] Efron B., R. J. Tibshirani. “An introduction to the bootstrap“, Chapman & Hall;CRC 1998.

[83] Panasyuk, Aleksey, Kishan G. Mehrotra, and Edmund Szu-Li Yu. “Automated Location-Aware Influencer Evaluation.“ Proceedings of the 3rd International Conference on Vision, Image and Signal Processing. 2019.

[84] Meeder, Brendan, et al. “We know who you followed last summer: inferring social link creation times in twitter.“ Proceedings of the 20th international conference on World wide web. ACM, 2011.

[85] H. Kwak, H. Chun, and S. Moon. Fragile online relationship: a first look at unfollow dynamics in Twitter. In Proc. of ACM CHI’11, Vancouver, BC, Canada, 2011.

[86] Lau, Jey Han, et al. “End-to-end network for twitter geolocation prediction and hashing.“ arXiv preprint arXiv:1710.04802 (2017).

[87] Ebrahimi, Mohammad, et al. “A unified neural network model for geolocating twitter users.“ Proceedings of the 22nd Conference on Computational Natural Language Learning. 2018.

[88] Zannettou, Savvas, et al. “Disinformation warfare: Understanding state-sponsored trolls on Twitter and their influence on the web.“ Companion proceedings of the 2019 world wide web conference. 2019.

[89] Rehurek, Radim. “Subspace tracking for latent semantic analysis.“ European Conference on Information Retrieval. Springer, Berlin, Heidelberg, 2011.

[90] Nguyen, Dong, et al. ““ How old do you think I am¿‘ A study of language and age in Twitter.“ Seventh International AAAI Conference on Weblogs and Social Media. 2013.

[91] Burger, John D., et al. “Discriminating gender on Twitter.“ Proceedings of the conference on empirical methods in natural language processing. Association for Computational Linguistics, 2011.

[92] Hargittai, Eszter, and Eden Litt. “The tweet smell of celebrity success: Explaining variation in Twitter adoption among a diverse group of young adults.“ New media & society 13.5 (2011): 824-842.

[93] Li, Linna, Michael F. Goodchild, and Bo Xu. “Spatial, temporal, and socioeconomic patterns in the use of Twitter and Flickr.“ cartography and geographic information science 40.2 (2013): 61-77.

[94] Flekova, Lucie, Daniel Preoţiuc-Pietro, and Lyle Ungar. “Exploring stylistic variation with age and income on twitter.“ Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). Vol. 2. 2016.

[95] Chang, Jonathan, et al. “ePluribus: Ethnicity on Social Networks.“ ICWSM 10 (2010): 18-25.

[96] Ciot, Morgane, Morgan Sonderegger, and Derek Ruths. “Gender inference of Twitter users in non-English contexts.“ Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. 2013.

[97] Culotta, Aron, Nirmal Ravi Kumar, and Jennifer Cutler. “Predicting the Demographics of Twitter Users from Website Traffic Data.“ AAAI. 2015.

[98] Golbeck, Jennifer, and Derek Hansen. “A method for computing political preference among Twitter followers.“ Social Networks 36 (2014): 177-184.

[99] Mohammady, Ehsan, and Aron Culotta. “Using county demographics to infer attributes of twitter users.“ Proceedings of the joint workshop on social dynamics and personal attributes in social media. 2014.

[100] Nechaev, Yaroslav, Francesco Corcoglioniti, and Claudio Giuliano. “SocialLink: exploiting graph embeddings to link DBpedia entities to Twitter profiles.“ Progress in Artificial Intelligence 7.4 (2018): 251-272.

Vita

Author's Name: Aleksey Valeriy Panasyuk

Place of Birth: Minsk, Belarus

Date of Birth: December 4, 1985

Degrees Awarded:

Masters of Science in Computer Science, SUNY Polytechnic Institute, 2011

Bachelor of Science in Computer Science, SUNY Polytechnic Institute, 2007

Bachelor of Science in Applied Mathematics, SUNY Polytechnic Institute, 2007

Professional Experience:

Associate Computer Scientist, Air Force Research Laboratory, May 2008 - present

Software Intern, Knowledge Intersect, June 2007 - June 2008