Language independent Big-Data system for the prediction of user location on Twitter

Jaime Alonso-Lorenzo, Enrique Costa-Montenegro, Milagros Fernández-Gavilanes
Telematics Engineering Department
University of Vigo, Spain
Email: jaimealonsolorenzo@gmail.com, {kike,milagros.fernandez}@gti.uvigo.es

***Abstract*—Social media interactions have become increasingly important in today's world. A survey conducted in 2014 among adult Americans found that a majority of those surveyed use at least one social media site. Twitter, in particular, serves 310 million active users on a monthly basis, and thousands of tweets are published every second. The public nature of this data makes it a prime candidate for data mining.**

Twitter users publish 140-character long messages and have the ability to geo-tag these tweets using a variety of methods: GPS coordinates, IP geolocation and user-declared location. However, few users disclose their location, only between 1% and 3% of users provide location data, according to our empirical findings.

In this article, we aim to aggregate information from different sources to provide an estimation on the location of any Twitter user. We use an hybrid approach, using techniques in the fields of Natural Language Processing and network theory. Tests have been conducted on two datasets, inferring the location of each individual user and then comparing it against the actual known location of users with geolocation information. The estimation error is the distance in kilometers between the estimation and the actual location. Furthermore, there is a comparison of the relative average error per country, to account for difference in country sizes. Our results improve those presented in different researches in the literature. Our research has as feature to be independent of the language used by the user, while most of works in the literature use just one language or a reduced set of languages. The article also showcases the evolution of our estimation approach and the impact that the modifications had on the results.

***Keywords*-Big-data; Social networks; Twitter; User location; Natural Language Processing; Network theory**

I. Introduction

Social networking sites are a pervasive part of modern life, with a majority of Internet users using them [1]. These sites work on the basis of relationships between people. These virtual connections often reflect real life connections: people tend to communicate with other people they may know in person. In the case of Twitter, a user follows other users, which in turn can follow the user back. A Twitter user tweets by publishing short text messages that are usually available publicly. As one of the leaders in social media, it is accessed by 310 million monthly users [2], and it is a massive source of global information. Consequently, Twitter has repeatedly been the subject of data mining research across a variety of domains: real-time event detection [3], stock market predictions [4], sentiment analysis [5], and even election prediction [6]. The data that such amount of simultaneous users generate at any given time is not unlike a global sensor network. We can use this information to gain new insights on human relationships and current events.

One of the most common applications of Twitter data is geolocation research. Tweets have been used to infer users' locations, travel patterns and disease spread. In spite of having a huge amount of data to conduct research on, there is a major hurdle: the scarcity of reliable, widespread location information in Twitter.

While Twitter has supported tweet geotagging since 2009 [7], penetration among users remains low. This may be explained by the opt-in nature of the feature. Twitter also provides other method of location: the free-text location field on a user's profile. But users are free to write anything on this location field, making this source inherently noisy. Also, far more users opt to disclose location information this way, but it does not change over time, unlike GPS and IP geolocation tags. In any case, this free-text location field is still meaningful, as it can act as a baseline for the location estimation of a user. Moreover, this information can be aggregated among a large group of users, by taking advantage of users' relationships.

With the rise of Twitter and similar platforms, vast amounts of new types of information have been made public to researchers. The concept of *Big Data* has emerged as a way to address these new challenges and many others. *Big Data* is a catch-all term for big datasets that are too large to be processed by traditional methods. As a part of this, non-relational databases (also known as NoSQL) have experienced a rise in popularity because they tackle many of the inadequacies that traditional relational databases suffer against big datasets, such as requirements for massive parallelization and fast and efficient handling of unstructured data. In this article, we choose a NoSQL document-oriented database, MongoDB[1], to facilitate the handling of big amounts of unstructured information extracted from Twitter and other sources.

Our approach leverages this information to form a novel and simple estimation algorithm, using Natural Language Processing (NLP) and network theory techniques, that improves upon previous results found in the literature, with the

[1]https://www.mongodb.com

978-1-4673-9005-7/16/$31.00 ©2016 IEEE

rare feature of being independent of the language used by the user.

This paper is organized as follows. We revise the state of the art in social networks location in Section II. In Section III, we detail the datasets and the data retrieval process used. Then, in Section IV, we explain the methodology and the evolution of the algorithms employed, to present an evaluation of the results according to different metrics in Section V. We introduce a brief discussion in Section VI, to finally conclude in Section VII.

II. Related work

Location estimation in social networks in general has received growing interest in academic research across the world. These researches typically use different techniques in the field of Natural Language Processing and network theory.

Regarding NLP techniques, this processing is used to extract semantic information from users' tweets that can be matched to geographic locations. Sadilek et al. [8] use language analysis to provide a layered approach that models user relationships and then user location by taking friendships into account. This method focuses on predicting locations in specific time windows. Their evaluation method is, however, highly localized in population-dense areas like New York City and Los Angeles. They benefit from the fact that this population-dense areas have a higher number of geo-active users (users that activate geolocation in their Twitter profile). Our method can be applied to any user regardless of location.

Cheng et al. [9] rely heavily on content and language analysis. Their approach does not use relationship modeling. Instead, they built a probability model for a set of words, using their geographical frequency to match those words to a set of U.S. cities. They achieve an accuracy of 51% of Twitter users place within 100 miles of their actual location, but this is also a localized algorithm that requires extensive preprocessing on specific locations (U.S cities with more than 5,000 inhabitants).

Mahmud et al. [10] improve on this work by implementing classifiers based on content-based features (words, hashtags, and location names). They also add another layer of information by incorporating time-zone analysis into their method.

Other authors such as Graham et al. [11] choose to compare several language recognition algorithms and apply them to a small set of cities in order to select the best estimation algorithm.

Some research has also been conducted using the free-text location field, like we do in this research. In the work presented by Hecht et al. [12], they analyzed the validity and completeness of the location information provided by Twitter users. They built classifiers based on this information and analyzed the results in a sample of users from four different countries. In this article, we present a method that is also based on the free-text location field that Twitter offers, achieving comparable or better results with less information, and not restricted to only four countries but worldwide.

Language-agnostic approaches have also been studied, involving mainly network theory techniques. These try to model a user network and their *following* relationships to estimate their location from an aggregation of data.

In [13], the authors implement a voting scheme that takes into account the GPS-tagged tweets of a user's friends to determine the user's location. Accuracy varies depending on the user and their number of geotagging friends. They test their algorithm using different sources of location information (GPS, declared location and GeoIP) and test their results on a group of users messaging about dengue fever.

Compton et al. [14] present an approach that also models a user network by establishing friendship relationships based on the people that a user mentions in their tweets. They also filter the Twitter accounts by removing non-human accounts that may retweet geo-tagged information, and accounts with a lot of location variation (i.e. people who travel). The results are based on the distance between friends in Twitter, using only geo-tagged data. They compare different executions of their algorithm by limiting the geographic dispersion of the user's network. They find that taking into account users with a geographically close network of friends minimizes their average and median error.

Conversely, our method provides a language-agnostic location estimation for users with no underlying assumptions. We simplify the network model of a user by only taking into account the accounts that are followed by the said user, and use the free-text location field to perform our estimations. While the average error varies in different countries, the estimations have been made on a global scale. In Section VI, we will show that we outperform several of the estimation algorithms in the literature.

III. Dataset collection

We used Twitter's Streaming API[2] in order to retrieve our tweets' datasets. Nearly 3,000,000 tweets were retrieved between the months of December 2015 and April 2016. These tweets were split in two different datasets, as shown in Table I.

Table I
Dataset Statistics

	Dataset 1	**Dataset 2**
Tweets	1,622,143	1,264,325
Geolocated tweets	30,081 (1.85%)	20,639 (1.63%)
Users	653,370	1,057,532
Geolocated users	18,382 (2.81%)	19,619 (1.85%)

The first dataset was retrieved in the days and weeks leading up to the Spanish general election on December

[2]https://dev.twitter.com/streaming/overview

20th, 2015, and the tweets were obtained by filtering the tweet stream on election-related keywords. Therefore, most of these tweets are in Spanish and most of the users were located in Spanish-speaking countries. The full country breakdown is detailed in Table II.

Table II
GEOGRAPHIC DISTRIBUTION: DATASET 1

Country	Users	Percentage
Spain	5,976	32.5%
Argentina	2,462	13.4%
Brazil	1,976	10.7%
France	1,100	5.98%
United States	1,050	5.71%
Venezuela	919	4.99%
Mexico	708	3.85%
Colombia	461	2.50%
Uruguay	310	1.69%
Other	3,420	18.6%

The second dataset was retrieved between the months of February 2016 and April 2016, and contains tweets from an unfiltered sample of the general Twitter stream. This data was retrieved to evaluate the estimation algorithm on a global dataset and avoid the possible bias that a filtered dataset could introduce. Table III provides the user distribution per country in this dataset.

Table III
GEOGRAPHIC DISTRIBUTION: DATASET 2

Country	Users	Percentage
United States	2,861	14.6%
Japan	2,142	10.9%
United Kingdom	1,321	6.73%
Brazil	1,292	6.58%
Philippines	1,165	5.94%
Indonesia	1,025	5.22%
Malaysia	962	4.90%
Argentina	716	3.65%
Thailand	620	3.16%
Other	7,515	38.3%

The actual data that was used by the estimation algorithm was the set of users with geolocation information. This list was extracted from each dataset, obtaining 18,382 users from the former dataset and 19,619 users from the latter.

Twitter's Streaming API was accessed using Twitter4J[3] in Java and Tweepy[4] in Python. Tweets were stored in a non-relational MongoDB database. The database was accessed and queried using MongoDB's native Java and Python APIs. MongoDB was chosen due to its easy integration with JSON documents (which are the default output of Twitter's Streaming API, schema-less model that allowed us to store data with different attributes according to our requirements, and MongoDB's aggregation framework that allowed us to deploy queries easily to extract information and insights from the data.

[3]http://twitter4j.org
[4]http://tweepy.readthedocs.io

IV. METHODOLOGY

Once we had the datasets with enough Twitter users, we established a baseline algorithm that provided a simple estimation of the location of a user. Afterwards, several iterations were designed, implemented and compared against one another. In this section, we will explain the methodology that was followed in the design process of every iteration of the estimation algorithm. Every iteration was then evaluated against the baseline algorithm. Our different proposed algorithms use both NLP and network theory techniques.

One of the main sources of information that were used in our NLP techniques was DBpedia[5]. DBpedia is a project aimed at serving structured data extracted from Wikipedia. As part of the Linked Data movement, it provides a way of presenting information that can be queried semantically. It also links concepts to other similarly structured datasets. This data is accessed using a semantic query language, such as SPARQL[6]. Most semantic data sources provide a SPARQL endpoint, like DBpedia[7], which is accessed by applications to extract data.

We chose DBpedia, which was used mainly to detect places, because it has information in more than 100 languages, like Wikipedia. DBpedia was used in every iteration to detect whether a string of characters in the Twitter's free-text location field described a place or not. Another use of DBpedia was to infer to what group popular Twitter accounts belong to. We queried DBpedia in each user's self-declared language on Twitter; if DBpedia was not available in a certain language, our algorithm defaulted back to English.

All of the iterations use a similar network model, based on the assumption that a user in Twitter follows their friends and/or people and organizations close to them.

The main data source for the estimation process was the free-text location field in their friends' profiles, that was analyzed using DBpedia to search for toponyms. These two components (network model and free-text location field analysis) were altered in different ways in every iteration.

For every iteration of the algorithm, the Twitter Streaming API was accessed to retrieve information for every user and the user's friends in JSON format. This information was cached in a MongoDB collection to prevent redundant queries in subsequent executions.

We used different algorithms that will be explained later in this section. The results obtained by the different algorithms

[5]http://dbpedia.org
[6]http://www.w3.org/TR/rdf-sparql-query/
[7]http://dbpedia.org/sparql

in every iteration is a list of places with weights assigned to them. For evaluation purposes, the chosen metric was the distance in kilometers between the inferred location (the location with the highest weight) and the actual location (GPS coordinates). This distance was calculated using the Haversine formula, which calculates the distance in a straight line between two sets of coordinates in a sphere. Let (ϕ_1, λ_1) be the set of latitude and longitude coordinates of point 1, (ϕ_2, λ_2) the set of latitude and longitude coordinates of point 2, $\mathrm{hav}(\theta) = \sin^2\left(\frac{\theta}{2}\right)$, and $r = 6367 \ \mathrm{km}$ (an approximation of the radius of Earth), the Haversine formula is defined as follows:

$$d = 2r \arcsin\left(\sqrt{\mathrm{hav}(\phi_2 - \phi_1) + \cos(\phi_1)\cos(\phi_2)\mathrm{hav}(\lambda_2 - \lambda_1)}\right)$$

The estimation for every user was represented in a JSON string containing the list of places with weights, the inferred location, their username, and the distance in kilometers with their actual location. We can see an example in the next JSON string:

```
{"weight_list": [["Madrid", 0.357], ["
   Valencia", 0.214], ["Sevilla", 0.214]
   , ["Spain", 0.142], ["Vigo", 0.071]],
    "inferred_location": ["Madrid"], "
   screen_name": "username", "error_km":
    [305.99]}
```

For the user represented by this JSON string, the inferred location is Madrid (the location with the highest weight), but the actual location is Valencia. The error in kilometers is 305.99, the distance in a straight line between those two cities. When the inferred location was a country, we calculated the distance between the actual location and the geographic center of the inferred country.

Now we will explain the different algorithms used and, in Section V, we will explore further the evaluation of the algorithms by performing several tests and comparisons.

A. Baseline algorithm

This estimation was calculated using a simple model of the user's network. Due to Twitter's API rate limits, we retrieved up to 200 of the accounts followed by each of the analyzed users.

The free-text location field of every one of these accounts followed by the user was gathered and then analyzed. In order to retrieve possible place names from these fields, they were split up into words, bigrams, and trigrams. Then, DBpedia was queried with each of these terms to check if they matched with an actual location on Earth. This was accomplished by checking if DBpedia had an entry with that term's name, and verifying that the entry had the `http://dbpedia.org/ontology/Place` property. The terms that passed these two conditions were added to a list.

Let us look into a practical example. Take a user with "New York City" as his/her declared free-text location field. This string of characters is divided into words, bigrams and trigrams. The word array would be {*"New"*, *"York"*, *"City"*}, the bigram array would be {*"New York"*, *"York City"*}, and the trigram array would be {*"New York City"*}. Each of these terms is searched in DBpedia to verify if it describes a location on Earth. If it does, it is added to the global list of possible places extracted from every user. The global list is implemented with a hashtable that keeps track of the number of occurrences of each term in the dataset (their frequency).

After this step, the simple baseline estimation is a list of locations with weights assigned to them. This list is the ordered list of terms by frequency, extracted from the hashtable. The twenty terms with the highest frequency are selected, and the weights are calculated by normalizing these frequencies so they all add to one.

B. First weighted version

The second iteration of the location estimation algorithm is based on the baseline algorithm. In this version, accounts belonging to organizations are weighed differently than personal accounts. This stems from the assumption that a user in Twitter follows organizations (newspapers, local police or fire departments, etc.) close to their location.

DBpedia was used in order to infer whether an account belonged to the organization or the non-organization category. In the account retrieval process, we selected the most popular accounts. Popular accounts must have a high number of followers, significantly more followers than accounts followed, and have a verified status on Twitter. In our tests, we selected Twitter accounts with 1,000 followers or more, and with a follower/following ratio of 100 or more (the account needed to be followed by at least 100 times more users than the amount of users the account is following).

Once an account was deemed to be popular, their user name was used to establish whether the account belong to an organization or not, by checking if their DBpedia entry had the `http://dbpedia.org/ontology/Person` property. After this, the same general approach was followed: place names were extracted from their free-text location fields. The main difference in this case is that the data gathered from organizations was weighed more than the rest. This was accomplished by modifying the hashtable where the terms and the occurrences were stored. The location terms extracted from a popular account were assigned a weight of 2, while the rest were assigned a weight of 1. After this, we followed the same procedure of sorting the hashtable by frequency and extracting the twenty highest weighted terms to perform the evaluation.

C. Second weighted version

Following the same line of thought that we used to get to the previous algorithm, we classified accounts in three categories: friends, celebrities, and organizations. Accounts were classified as friends if they were not popular; celebrities if they were popular and their DBpedia entry had the `http://dbpedia.org/ontology/Person` property; and organizations if they were popular and their DBpedia entry did not have the `http://dbpedia.org/ontology/Person` property. Accounts were deemed to be popular according to the same attributes than before.

Once every account was classified into one of these categories, they were assigned weights in terms of percentage, e.g. 60% to the friends category, 10% to celebrities and 30% to organizations. This meant that an account in category i was given a weight of W_i/N_i, W_i being the weight assigned to category i (60%, 10% and 30%), and N_i being the number of accounts that belonged to that category.

These percentages were selected after various tests, as they were the ones that provided better results against the baseline algorithm. The result was also given in terms of a list of weighted locations.

D. Third weighted version

The two previous iterations of the algorithms weigh every account in a category the same way. However, the contribution of each account should not carry the same weight. A mildly popular celebrity account (e.g. a regional star) followed by a user is more meaningful that a very popular and international celebrity account followed by that user. To take this into account, we modified the algorithm to weigh every popular account (celebrities and organizations) following an inverse relationship with their number of followers. Thus, extremely popular accounts would receive a lower weight than the mildly popular ones.

The weight of an account in this version was obtained by multiplying the total weight assigned to their class (30% for organizations, and 10% for celebrities) by the inverse relationship between the account followers and the total followers in their class. This meant that higher weights were assigned to accounts with few followers, and lower weights were assigned to accounts with many followers. This inverse relationship was only applied to popular accounts (i.e. celebrities and organization) and was not applied to friends (obscure accounts).

E. Layered version

Previous algorithms changed the weight of the accounts in the obtained list of locations. With this algorithm we want to improve the locations once the list is already obtained. This means that we can obtain results using the previous algorithms and then deciding to use this one or not.

In the evaluation process, we noticed that a number of users were assigned countries as their most likely location, while their actual, more specific location had a lower weight. Usually, their actual location was a specific place (city, region or state) within that country. In this version of the estimation algorithm, we started from any of the previous algorithms and modified the weight list afterwards.

To account for the unspecificity of the results, we used DBpedia to differentiate the information in the free-text location field between countries and non-countries (by looking for the `http://dbpedia.org/ontology/Country` property). Moreover, cities, regions and states in DBpedia are usually linked to their country, and this was used to establish relationships between places (cities, regions and states with their respective countries).

Cities, regions and states were added the weight assigned to the countries that they belonged to, if they appeared on the weight list. This was carried out to narrow down the final estimations by choosing more specific locations.

For example, take a user (located in Madrid) with the following weight list:

```
{"weight_list": [["Spain",0.4], ["Madrid
    ",0.25], ["Barcelona",0.15], ["Paris"
    ,0.1], ["Cadiz",0.1]] }
```

In this case, Spain is the inferred location, with a weight of 0.4. However, the user's actual location, Madrid appears in the weight list, but it was not assigned the highest weight. Thanks to DBpedia, we can verify that several of those locations are cities located in Spain (Madrid, Barcelona and Cadiz). After adding Spain's weight to the cities in Spain, we have the following weight list:

```
{"weight_list": [["Madrid",0.65], ["
    Barcelona",0.55], ["Cadiz",0.5], ["
    Spain",0.4], ["Paris",0.1]] }
```

And in this case, the inferred location would match the user's actual location.

V. Evaluation

As already mentioned in the previous section, the main evaluation metric was the distance in kilometers between the estimation and the actual location, using the Haversine formula. First, we would calculate the average and the median error of this distance for a sample of users.

We also compared every result to the corresponding result in the baseline algorithm, and we also show this difference in terms of percentage, to compare when each algorithm performs better.

Additionally, we conducted a country-level verification and calculated the accuracy of this metric.

In order to evaluate the results, we extracted a random sample of 500 users from dataset 1 and another sample of

500 users from dataset 2. Every algorithm was tested on the same two samples of users, and numerical metrics were calculated using NumPy[8].

Tables IV and V show the results for the different algorithms in charge of the weight assigned to the locations of the accounts followed by the user. i.e. baseline, first weighted, second weighted and third weighted. Those results are calculated applying or not to each of the previous algorithms the layered algorithm to improve the locations on the list.

The results include the average error and the median error, all expressed in kilometers. They also include the country-level accuracy, i.e. the percentage of user for whom we have obtained their country correctly.

We can see that the average error is significantly higher in the second dataset. This may be caused by the unreliability of DBpedia's data in some languages (which is extracted from the corresponding versions of Wikipedia). Most of the data in dataset 1 was in Spanish, whereas dataset 2 contains a significant amount of data in Japanese, Malay, or Tagalog, which have smaller versions of DBpedia.

We have also computed a country-level metric to find out the percentage of users for which the estimated location was at least located within their actual country. Some of the users could not be evaluated due to missing or inconsistent data in DBpedia. The percentage of these type of users was also calculated. The results for the two datasets and different algorithms are displayed in Table VI. This table shows, as well as the results in Tables IV and V, that, while the error in kilometers may be relatively high in some cases, our estimation algorithm locates a majority of users within their actual countries. These results in Table VI also prove that dataset 2 has higher country-level errors as higher percentage of users cannot be evaluated due to missing or inconsistent data in DBpedia for not so popular languages.

The result for the first weighted version, which uses a weight of 2 for organization accounts and 1 for the rest, shows that both the average and median errors are higher than the baseline algorithm's errors. Despite the average and relative error metrics being generally worse than the baseline algorithm's metrics, the country-level accuracy improves in dataset 1, while remaining almost the same in dataset 2. This could suggest that, in general, a user mainly follows organizations based in their home country, improving our country-level estimation, while affecting negatively the estimation's average error. The median error is also considerably lower than the average error, which can mean that a few outliers are responsible for the increase in the average error, but 50% of the users are still located within 300 kilometers of their location.

In the second weighted version, we performed the same tests on a dataset separated over three categories, friends, celebrities, and organizations, weighed 60%, 10%, and 30%, respectively. The results shows that the average error in dataset 1 is fairly higher than what the two previous tests showed. On the other hand, the error in dataset 2 is comparable to the previous weighted algorithm. A possible explanation for this is that splitting users into categories and distributing weights to them may cause situations in which a few accounts have too much of an influence on the final result.

The third weighted version used the same weight distribution than the previous version (60% for friends, 10% for friends, and 30% for organizations), but making each weight inversely proportional to each user's number of followers. This assigns higher weights to locations extracted from more obscure accounts, and lower weights to popular accounts. This was only added for relatively popular accounts, i.e. celebrities and organizations. The results show the highest errors so far, compared to all the previous iterations. This may be an argument against the assumption of the amount of influence that more obscure accounts have over more popular ones. They also show lower country-level accuracy in the estimations.

The baseline and layered version is based on the baseline algorithm, with a modification on the weight list to group cities and their countries. It is a different approach than what we have put forward so far, by manipulating the weight list instead of changing the network model. According to our findings, the layered version presents promising improvements over the original baseline algorithm. While the average error is somewhat higher for both datasets, the median error has decreased 51% in dataset 1 and 18% in dataset 2, compared to the baseline algorithm. The country-level accuracy remains in a similar level compared to the baseline algorithm. Due to the improvement that the layered algorithm showed, we decided to test it on the weighted algorithms as well. The average error improves in some cases, even achieving the best result so far in the first weighted and layered algorithm, and the median error improves in every case, like we have seen with the baseline and layered algorithm. Another interesting result is the country-level accuracy achieved by combining the first weighted and layered algorithm in dataset 1, which is 93.2%, the highest so far.

Tables VII and VIII show the results of the average error for the different algorithms (baseline, first weighted, second weighted, third weighted and baseline and layered) for the nine countries with more users in each of the datasets. The results include the average error expressed in kilometers, and also, in brackets, the difference between the current algorithm and the baseline one. These were calculated by averaging the error among the users whose actual location was in one of those countries.

With the first weighted algorithm, the average error per country is generally higher for both datasets. Six out of

[8]http://www.numpy.org

Table IV
RESULTS FOR THE DIFFERENT ALGORITHMS WITH DATASET 1

	Baseline	1st W.	2nd W.	3rd W.	B. & L.	1st W. & L.	2nd W. & L.	3rd W. & L.
Average error (km)	676.76	694.95	860.64	929.46	695.06	**658.93**	851.42	987.40
Median error (km)	287.80	313.83	325.61	341.29	**141.04**	235.11	255.50	312.48
Country-level accuracy	89.6%	92.8%	90.6%	87.6%	90.6%	**93.2%**	89.8%	88.2%

Table V
RESULTS FOR THE DIFFERENT ALGORITHMS WITH DATASET 2

	Baseline	1st W.	2nd W.	3rd W.	B. & L.	1st W. & L.	2nd W. & L.	3rd W. & L.
Average error (km)	**1034.31**	1204.05	1200.47	1395.55	1198.92	1200.77	1209.37	1289.38
Median error (km)	226.64	283.92	282.60	307.28	**185.53**	201.39	208.08	237.45
Country-level accuracy	83.6%	83.4%	**85.4%**	78.4%	83.8%	83.8%	84%	81.8%

Table VI
COUNTRY-LEVEL METRIC FOR DIFFERENT ALGORITHMS

		B.	1st W.	2nd W.	3rd W.	B. & L.
D1	Correct	89.6%	92.8%	90.6%	87.6%	90.6%
	Incorrect	6.8%	1.8%	7.8%	10.4%	7.2%
	Error	3.6%	5.4%	1.6%	2%	2.2%
D2	Correct	83.6%	83.4%	85.4%	78.4%	83.8%
	Incorrect	13%	12%	10.4%	14.4%	13.2%
	Error	3.4%	4.6%	4.2%	7.2%	3%

the nine analyzed countries in dataset 1 and dataset 2 have higher error compared to the baseline algorithm. We show the difference of each result and the baseline algorithm's result in terms of percentage. With the second weighted version, similar results were obtained when the error per country was calculated, for most countries, the average error was much higher, compared to the original baseline algorithm. Even though most predictions are further off than before, the country-level accuracy shows that they still remain in the correct country. The third weighted version show similar results, the estimations are worse for most countries, compared to the baseline algorithm. But with the baseline and layered version, the error per country improves, especially in dataset 2, which shows improvements for six out the nine analyzed countries.

Nevertheless, we found that the error was dependent on the user's country. Location estimation of users in bigger countries yields, in general, a higher error in absolute terms. That is why we also decided to calculate a relative metric of error. This was calculated by dividing the absolute error by the square root of the country's area (supposing a square shape).

Let μ_i be the average error for country i, and let a_i be the surface area of country i, the relative error is defined as follows:

$$\text{Relative error } (\%) = \frac{\mu_i}{\sqrt{a_i}} \cdot 100$$

To illustrate the previous point of higher errors in bigger countries, we present the relative error in percentages of the countries with more users in the two datasets in Tables IX and X with the baseline algorithm. These results show an acceptable relative error in most of the countries except in the long-shaped countries (such as Japan, United Kingdom, Philippines and Malaysia), i.e. with a shape very different from the supposed square shape.

Figures 1 and 2 show the error distribution among the two datasets and the different algorithms. Both graphs have a cutoff of 1,000 kilometers, because most estimations have a lower error (between 80% and 90% depending on the algorithm, exact figures can be seen in Table XI). Most users are located within 400 kilometers of their actual location with every algorithm, like we verified with the median error. The difference in median errors becomes apparent in the baseline and layered algorithm: 40% of users (resp. 35%) are located with a maximum error of 50 kilometers in dataset 1 (resp. dataset 2).

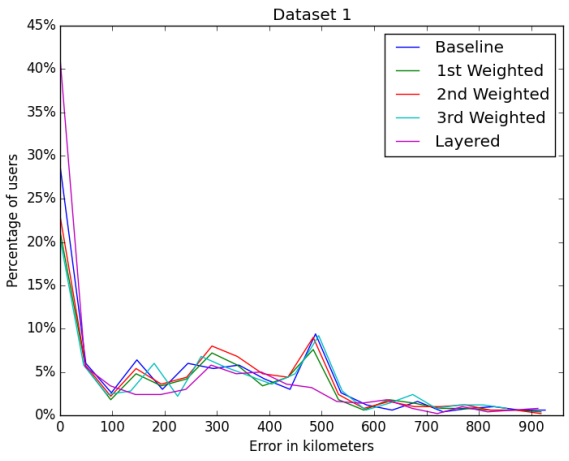

Figure 1. Error distribution graphs (Dataset 1)

As a conclusion, we can state that the results are con-

Table VII
AVERAGE ERROR (KM) PER COUNTRY COMPARISON WITH DATASET 1

Country	Baseline	1st Weight	2nd Weight	3rd Weight	Baseline & Layered
Spain	314.61	502.66 (+59.77%)	687.59 (+118.55%)	657.93 (+109.13%)	339.55 (+7.93%)
Argentina	504.16	599.88 (+18.99%)	603.43 (+19.69%)	671.91 (+33.27%)	324.30 (-35.68%)
Brazil	784.75	1620.8 (+106.54%)	3508.55 (+347.09%)	2257.35 (+187.65%)	631.01 (-19.59%)
France	254.84	305.9 (+20.04%)	511.63 (+100.77%)	274.78 (+7.82%)	381.45 (+49.68%)
US	1237.84	1125.3 (-9.09%)	2038.18 (+64.66%)	2145.92 (+73.36%)	2516.82 (+103.32%)
Venezuela	429.60	363.79 (-15.32%)	903.19 (+110.24%)	1225.24 (+185.20%)	605.02 (+40.83%)
Mexico	759.61	1094.51 (+44.09%)	921.37 (+21.30%)	810.57 (+6.71%)	402.98 (-46.95%)
Colombia	440.77	2178.55 (+394.26%)	1204.72 (+173.32%)	1541.56 (+249.74%)	1190.81 (+170.17%)
Uruguay	934.44	180.6 (-80.67%)	180.65 (-80.67%)	213.91 (-77.11%)	112.15 (-88.00%)

Table VIII
AVERAGE ERROR (KM) PER COUNTRY COMPARISON WITH DATASET 2

Country	Baseline	1st Weight	2nd Weight	3rd Weight	Baseline & Layered
US	1071.43	1229.14 (+14.72%)	1098.47 (+2.52%)	1824.91 (+70.32%)	1061.70 (-0.91%)
Japan	962.79	1136.26 (+18.02%)	1327.33 (+37.86%)	340.46 (-64.64%)	863.71 (-10.29%)
UK	860.38	793.35 (-7.79%)	793.50 (-7.77%)	837.37 (-2.67%)	649.69 (-24.49%)
Brazil	2407.73	1636.01 (-32.05%)	1462.47 (-39.26%)	4708.13 (+95.54%)	4194.73 (+74.22%)
Philippines	901.49	559.98 (-37.88%)	880.34 (-2.35%)	1791.19 (+98.69%)	909.79 (+0.92%)
Indonesia	810.20	1043.60 (+28.81%)	905.98 (+11.82%)	878.05 (+8.37%)	111.64 (-86.22%)
Malaysia	1059.15	1206.82 (+13.94%)	1320.47 (+24.67%)	1554.32 (+46.75%)	330.63 (-68.78%)
Argentina	422.38	453.23 (+7.30%)	330.4 (-21.78%)	626.80 (+48.40%)	223.03 (-47.20%)
Thailand	393.08	405.11 (+3.06%)	448.66 (+14.14%)	709.99 (+80.62%)	411.19 (+4.61%)

Table IX
RELATIVE ERROR WITH BASELINE ALGORITHM IN DATASET 1

Country	Relative Error
Spain	44.2%
Argentina	30%
Brazil	26.9%
France	34%
United States	40%
Venezuela	44.9%
Mexico	54.2%
Colombia	41.2%
Uruguay	219.6%

Table X
RELATIVE ERROR WITH BASELINE ALGORITHM IN DATASET 2

Country	Relative Error
United States	34.99%
Japan	156.6%
United Kingdom	174.57%
Brazil	82.5%
Philippines	154.1%
Indonesia	58.7%
Malaysia	184.1%
Argentina	25.33%
Thailand	54.9%

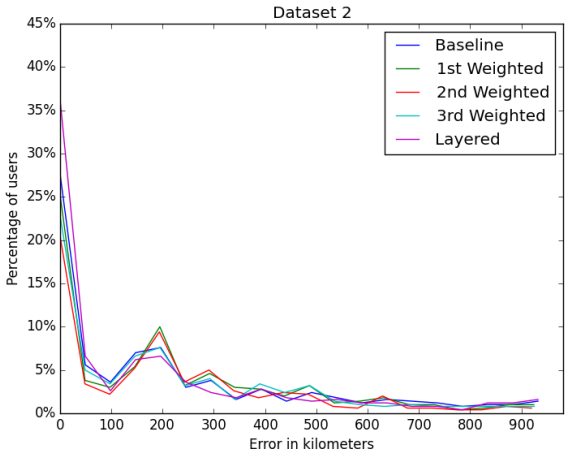

Figure 2. Error distribution graphs (Dataset 2)

sistently better in dataset 1, which had a majority of users from Spanish-speaking countries. Like we mentioned before, this may be caused by the unreliability of DBpedia in more obscure languages, whereas the Spanish DBpedia is one of the biggest.

The baseline algorithm performs better in terms of average error in both cases; but the baseline and layered algorithm

Table XI
PERCENTAGE OF USERS INSIDE THE ERROR DISTRIBUTION GRAPHS

	Baseline	1st Weighted	2nd Weighted	3rd Weighted	Baseline & Layered
Within 1000 km (Dataset 1)	89.8%	88.46%	86.8%	85.4%	89%
Within 1000 km (Dataset 2)	77.8%	75.6%	75.4%	71.4%	82%

performs best in terms of median error. On the other hand, the first weighted and layered algorithm has the highest country-level accuracy in dataset 1, and the second weighted algorithm has the highest country-level accuracy in dataset 2. Anyway, in almost all of the cases we obtain a country-level accuracy between 80% and 90%.

VI. DISCUSSION

In this section we compare our results with the ones in the researches in the state of the art.

We want to remark that our proposal works with users from any country of the world and it is also non language dependant, as it works with almost any of the most popular languages, as our system supports more than 100 languages. This is proven with the use of dataset 2 obtained by capturing an unfiltered sample of the general Twitter stream. Most of the proposals in the literature are centered in a country or language. For example Sadilek et al. [8] evaluation method is highly localized in population-dense areas, as they capture tweets from New York City and Los Angeles, benefiting from the fact that this population-dense areas have a higher number of geo-active users. Cheng et al. [9] and Mahmud et al. [10] only work in a set of cities as they built a probability model for a set of words, using their geographical frequency to match words to a set of U.S. cities with more than 5,000 inhabitants. Hecht et al. [12] only analyzed their results in a sample of users from four different countries. And Graham et al. [11] use several language recognition algorithms but applying them only to a small set of cities.

We can compare this results to past works evaluated in similar terms like Cheng et al. [9], that obtained a best result of 860 kilometers in average error, using a localized algorithm based on U.S. cities. We beat this result in most cases with dataset 1, which contained information from more than 15 countries. They also claimed to located a 51% of users within 100 miles (160.9 kilometers) of their actual location. We beat that on dataset 1 with the layered algorithm (median error of 141.04 kilometers), and achieve reasonable results in other cases, considering that we are using worldwide datasets.

Hecht et al. [12] also used the location field to perform estimations, obtaining a country-level accuracy of up to 88.86% in a dataset with users from four different countries. We also beat this result in most cases in dataset 1, and achieve a reasonably close result in dataset 2. But our datasets are not restricted to a given number of users.

To conclude, we presented a language independent location estimation method for users with no underlying assumptions. We use a simple network model of a user by only taking into account the accounts that are followed by said user, and use the free-text location field to perform our estimations. While the average error varies in different countries, the estimations have been made on a global scale.

VII. CONCLUSIONS AND FUTURE WORK

Location prediction on social networks is a recent topic of research, but it is linked to more established fields like NLP and geocoding. In this article, we focus on predicting home locations of Twitter users, with varying degrees of success depending on the algorithm and the user. Unlike many other past works, we use a relatively overlooked source of information for location prediction; self-declared user location; and achieve comparable or better results. This data source is noisy; but we find that, by aggregating enough results, a reasonable estimation can be reached. We achieved promising results in terms of country-level accuracy (80%+), and our average error distance also improved in most cases compared to previous works.

We have also showed the evolution of our estimation procedure. In order to improve the algorithm's accuracy, we implemented and studied several alternatives. Most of our efforts were placed on network modeling modifications, by classifying accounts in several categories and assigning weights to them.

In this article, we achieve results comparable or better to past works, with relatively little information. This algorithm can be improved upon in the future in two main fronts: network modeling and language analysis, which are the two major techniques used in the literature.

We use a network model based on the accounts followed by each user, but this can be improved by only taking into account those users with a reciprocal following relationship; that is, users that follow each other. This is a better indicator for a friendship relationship between users on Twitter and could be used to filter information. Geographical information can also be extracted from tweets, which can be aggregated across the most recent tweets by the user's friends. Some authors have also analyzed the time of the day that users tweet at to determine the time-zone of a user, which is also another indicator for their location.

Language analysis has been used in the past to match terms to locations. Named-Entity Recognition (NER) may be used to extract meaningful geographical information or

to detect toponyms on tweets that might give more insight on the user's location. Moreover, language and regional dialect detection would also be useful to improve estimations.

ACKNOWLEDGMENT

This work was partially supported by the Spanish Government, co-financed by the European Regional Development Fund (ERDF), under project TACTICA and by the Ministerio de Economía y Competitividad under project COINS (TEC2013-47016-C2-1-R).

REFERENCES

[1] Duggan, M., Ellison, N. B., Lampe, C., Lenhart, A., & Madden, M. (2015). Social media update 2014. Pew Research Center, 19.

[2] Twitter Inc. (2016). Twitter Company Site. Retrieved May 6, 2016, from https://about.twitter.com/company

[3] Sakaki, T., Okazaki, M., & Matsuo, Y. (2010, April). Earthquake shakes Twitter users: real-time event detection by social sensors. *In Proceedings of the 19th international conference on World wide web* (pp. 851-860). ACM.

[4] Bollen, J., Mao, H., & Zeng, X. (2011). Twitter mood predicts the stock market. Journal of Computational Science, 2(1), 1-8.

[5] Fernández-Gavilanes, M., Álvarez-López, T., Juncal-Martínez, J., Costa-Montenegro, E., & González-Castaño, F. J. (2016). Unsupervised method for sentiment analysis in online texts. *Expert Systems with Applications*, 58, 57-75.

[6] Tumasjan, A., Sprenger, T. O., Sandner, P. G., & Welpe, I. M. (2010). Predicting elections with twitter: What 140 characters reveal about political sentiment. ICWSM, 10, 178-185.

[7] I. P. (2009, November 20). Twitter Geotagging: What You Need to Know. Retrieved May 6, 2016, from http://tinyurl.com/yadlgfw

[8] Sadilek, A., Kautz, H., & Bigham, J. P. (2012, February). Finding your friends and following them to where you are. *In Proceedings of the fifth ACM international conference on Web search and data mining* (pp. 723-732). ACM.

[9] Cheng, Z., Caverlee, J., & Lee, K. (2010, October). You are where you tweet: a content-based approach to geo-locating twitter users. *In Proceedings of the 19th ACM international conference on Information and knowledge management* (pp. 759-768). ACM.

[10] Mahmud, J., Nichols, J., & Drews, C. (2012). Where Is This Tweet From? Inferring Home Locations of Twitter Users. *ICWSM*, 12, 511-514.

[11] Graham, M., Hale, S. A., & Gaffney, D. (2014). Where in the world are you? Geolocation and language identification in Twitter. *The Professional Geographer*, 66(4), 568-578.

[12] Hecht, B., Hong, L., Suh, B., & Chi, E. H. (2011, May). Tweets from Justin Bieber's heart: the dynamics of the location field in user profiles. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 237-246). ACM.

[13] Davis Jr, C. A., Pappa, G. L., de Oliveira, D. R. R., & de L Arcanjo, F. (2011). Inferring the location of twitter messages based on user relationships. *Transactions in GIS, 15(6)*, 735-751.

[14] Compton, R., Jurgens, D., & Allen, D. (2014, October). Geotagging one hundred million twitter accounts with total variation minimization. *In Big Data (Big Data), 2014 IEEE International Conference on* (pp. 393-401). IEEE.