IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 10, 2018, accepted May 27, 2018, date of publication June 4, 2018, date of current version June 20, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2843814

Geotagging Text Data on the Web–A Geometrical Approach

MANSI A. RADKE[1], NITIN GAUTAM[2], AKHIL TAMBI[3], UMESH A. DESHPANDE[1], AND ZAREEN SYED[4]

[1]Visvesvaraya National Institute of Technology, Nagpur 440010, India
[2]Fidelity National Information Services, Pune 411067, India
[3]Factset Systems India Pvt. Ltd., Hyderabad 500032, India
[4]IBM Research, New York City, NY, USA

Corresponding author: Mansi A. Radke (mansiaradke@gmail.com)

ABSTRACT Geotagging, the process of tagging of documents with geographical information, is an important issue in geographical information retrieval systems. In this paper, we propose a novel approach for geotagging of textual web documents. Existing techniques disambiguate place names occurring in a document individually first and then find its focus. As opposed to this, the proposed approach, considers all the place names together without disambiguating them individually. It is a heuristic-based three step process. The first step is hierarchical, the second one is geometric technique, and last step reports the final focus. The approach requires only a gazetteer and a named entity recognition tool for processing. It reports the geographical focus of a web page as a tuple (latitude and longitude). We have tested our algorithm on two data sets, namely, the Wikipedia and Open Directory Project (ODP) data sets. The proposed approach reports the focus correctly up to continent level for 97.07% of the Wikipedia pages and correctly up to country level for 95.57% of them. Moreover, for more than 60% of the documents, the error in the distance between the actual focus, and the reported focus is within 15 km, i.e., less than 10 miles for this data set. In addition, the median error is 8.17 km. The technique also performs very well on huge data sets like ODP with sample size of around 600 000 web pages. The proposed system can be used in any geographic information retrieval system where geographic meta-data of documents needs to be obtained for indexing and searching.

INDEX TERMS Disambiguation, evaluation, geographic information systems, geotagging, information retrieval.

I. INTRODUCTION

A lot of queries with place names are being launched by people using the internet search engines. Typical examples are ''(Book shops in London)'', ''(Tourist places near New York)'' etc. For efficient search with place names, documents are tagged with geographical metadata. The process of assigning focus or foci to documents is called geotagging.

Geotagging is an important component of a geographic search engine. The main application of geotagging is in the area of geographic or spatial indexing. Also, it helps in query expansion and geographic ranking. Further it aids in document classification too.

The approaches dealing with geotagging comprise of three basic steps - named entity extraction, disambiguation and focus calculation. Persons, places, or organizations are the named entities. Named entity extraction is done by using a named entity recognition (NER) tool, such as the Stanford NER [11] , Alchemy [1] etc. The disambiguation step refers to associating unique co-ordinates with a place name. Two types of ambiguities need to be addressed while disambiguation. There could be multiple places on the earth known by the same name. This is referred to as geo/geo ambiguity. On the other hand, geo/non-geo ambiguity is another type of ambiguity, which occurs when the name of a place could also be the name of a person, an organization or an object. For example, Turkey is a country and the name of a bird too. The third step of geotagging is the focus reporting or focus finding step in which the actual focus of the page is found out depending on the disambiguated place names.

Many existing approaches disambiguate (or ground) the place names of a document first and then apply the focus determination algorithm. As a result of this, if there is an error in the disambiguation step, it is likely to propagate to

the focus determination step too. This may possibly lead to incorrect results.

To avoid this issue, in our proposed approach, we do not ground the individual place names first, but consider all of them together. We use a combination of a hierarchical and a geometric approach with the application of a heuristic technique to disambiguate place names occurring in a web document. The heuristic is based on the observation that when there are mentions of multiple place names in a document, the smallest region encompassing all of them disambiguates each place name appropriately. We then assign an appropriate geotag to it. The proposed approach consists of three steps. The first step is hierarchical, in which, we find the country level focus of the web page. The output of this step is given to the geometrical step, in which, we perform disambiguation and focus finding together. The output of the geometrical step is given to the third step which finally reports the focus of the web page.

The proposed approach requires the access to a named entity recognizer and a gazetteer. It does not need any other information for processing. We have performed extensive experimentation with the proposed approach and found the technique to perform better than the baselines.

Thus we summarize the contributions of this paper as follows:

1) We propose a heuristic based novel technique for geotagging of textual data on the web. The technique performs disambiguation and focus finding together in contrast to the existing techniques which perform these in a pipeline fashion. The technique has three basic steps: the hierarchical step, the geometrical step and the focus reporting step.
2) For the Wikipedia dataset we observed the following:
   - In 97.02% of the cases, the focus reported is correct up to the continent level and in 95.57% of the cases the focus reported is correct up to the country level.
   - For more than 60% of the documents, the error in the distance between the actual focus and the reported focus is within 15 km (i.e. less than 10 miles).
   - Overall, the median error is 8.17 km.
3) For the large ODP dataset, in 93.07% of the cases the reported focus is correct up to the continent level and in 90.27% of the cases the reported focus is correct up to the country level. For ODP dataset we could not measure the exact distance as the focus of the pages was not reported. Only hierarchy of the web pages was known.

In the following section, we discuss the related work. In section III, we present the proposed approach. This is followed by the experimental results in section IV. Finally we conclude the paper and point out some directions for future work in this area in section V.

II. RELATED WORK

The first attempt of developing a geotagging mechanism for textual documents dates back to 1994 when Woodruff et al. [25] proposed a system GIPSY (Georeferenced information processing system), in which the documents are geotagged to aid indexing and retrieval in search. A seminal work in the area of geotagging is by Amitay et al. in 2004 where a system ''(Web-a-where)'' is proposed by Amitay *et al.* [6], [7]. The system considers locations mentioned in the text of the web page and assigns a confidence level to each such location. Using this confidence, they assign a final focus or foci to the web page with a focus determination algorithm. The authors compare their results with the tags or hierarchical classification provided by the editors of the ODP (Open Directory Project). They report a geographical focus correct up to the country level in 91% of the cases.

Wang et al. [23] proposed algorithms for three types of geographical information. The first type is the source geography, i.e. the place of the server which stores the document. It is named as provider location by the authors. The next type is the target geography, which contains the locations that the document mentions. The authors call this as the content location. The third type, called the serving location, indicates the geographical scope that a web resource can reach. Their techniques involve mining the hyperlink structures and user logs. Zhang et al. [26] proposed the georank algorithm inspired by the standard page rank algorithm in which the authors resolve the geo/geo ambiguity. Their other contributions include a heuristic algorithm to address geo/non-geo ambiguity and an algorithm to find the focused locations of the document.

The other approaches for geotagging include work by Martins and Silva [19], where they built a web linkage graph from the named entities. Their approach is based on an ontology for disambiguation and a graph ranking algorithm to reach to a final focus of the document. Leidner et al. [15] used the ''(minimality heuristic)'', to geotag text data which states that, when there are multiple place names mentioned in a document, the smallest region encompassing all of them disambiguates each place name.

Zong et al. [27] used a rule based disambiguation approach. The disambiguated output is fed to the focus determination step, which constructs a segment tree for each page. Locations are assigned to appropriate segments in the tree, and their scores are calculated by performing a depth first search over the tree and using certain heuristics. Finally, the place with the greatest score along with its segment is reported as the focus. The authors claim to achieve an accuracy of 86.8%. Laere et al. [14] georeference the Wikipedia documents using data from social media resources like Flickr and Twitter. They show that their approach is better than gazetteer based approaches like Yahoo! Placemaker and also better than approaches that perform language modelling trained on Wikipedia.

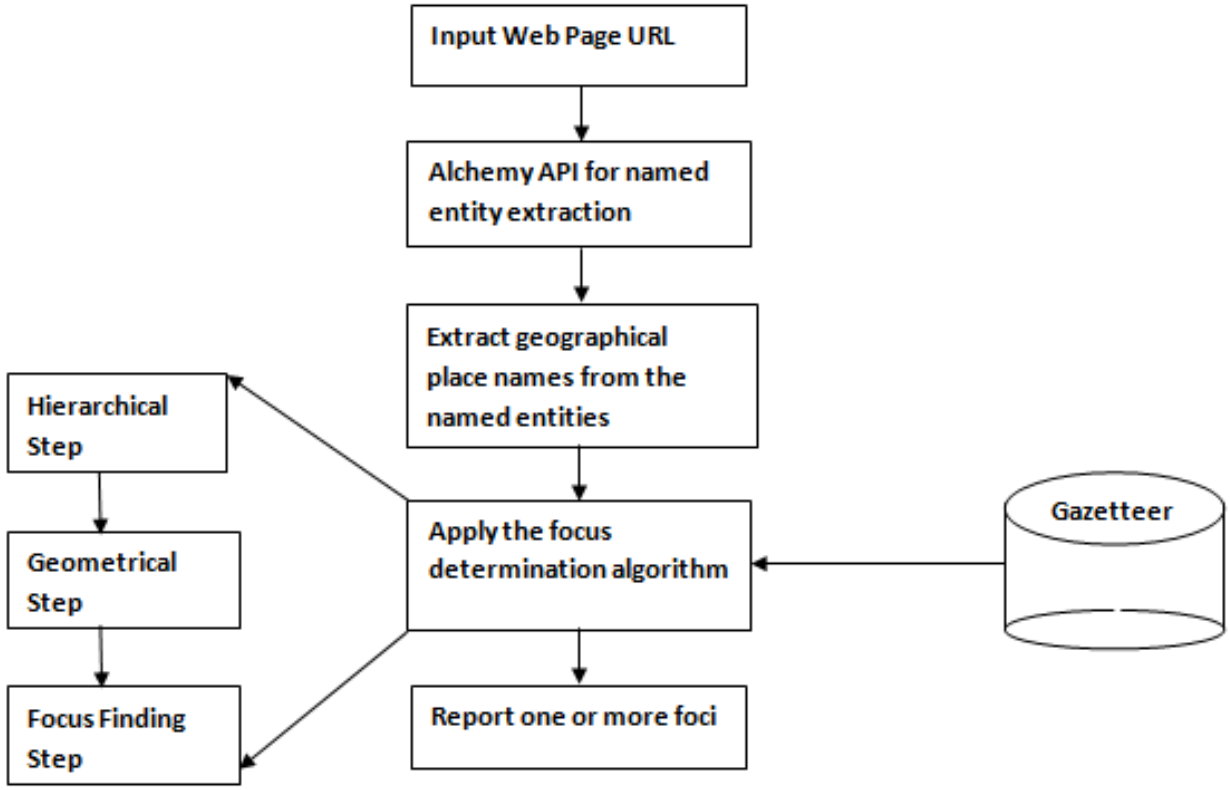

FIGURE 1. Block diagram of the proposed system.

In [10], the authors devised a simple method for finding the focus depending on the frequency of the place names in a document and evaluate their approach using Yahoo! Placespotter, Open Calais etc. They claim that they determine the focus of the news articles correctly up to country level in 95% of the cases.

Some other important pieces of work in this area are by Martins et al. [18], [19] and Andogah et al. [8].

Lieberman et al. [17] used proximity, sibling and prominence clues to geotag documents. Later in [16] they suggested the use of local lexicons in addition to global lexicons for geotagging to eventually facilitate spatial indexing. The authors consider the geotag to be correct if it lies within 10 miles of the ground truth value. They report their results in terms of precision, recall and F1. They report their results on self annotated data. Probabilistic language modelling approaches have also been considered for geotagging documents in [22] and [24]. Researchers have worked on Twitter data also, however we focus on web documents with substantial length. Min Chen et al. [9] have used place name patterns for finding the geographical focus of documents.

Some other recent approaches to geotagging are by Melo et al. where they have used hierarchical classifiers to geotag documents [20]. In [21] they have presented an extremely informative survey of geotagging methods. Lately language models and text mining is also used for geotagging. Additionally multimodal methods for georeferecing documents are gaining popularity where visual and textual features both are used to geotag the documents [12], [13].

In the next section, we explain the architecture of the proposed system for geotagging of web documents and explain its each component in detail.

III. PROPOSED APPROACH

The proposed approach is specifically designed for geotagging of textual data, especially web documents. We first preprocess the data by using the APIs provided by the Alchemy named entity extractor [1] to extract the place names. In the first step, i.e. the hierarchical step, we consider all the names occurring in the document, and find out their country level focus. We then extract all the entries of each place name from the gazetteer. In the second step, called the geometrical step, the extracted co-ordinates of the entries (also referred to as points) are placed onto a world map. Using various heuristics, some of the irrelevant points are pruned out. The last step, called the focus determination step, reports the focus (or foci) based on the points retained after the geometrical step. The block diagram of the proposed systems is as shown in the figure 1.

A. PREPROCESSING OF WEB DOCUMENTS

In this step, the web page uniform resource locator (URL) is given as input to Alchemy. The place names occurring in the web page content, along with the count of number

of occurrences of the place name, are extracted. These are referred to as locations. They are stored in a list L. Once this preprocessing step is done, the list L is given as input to the Hierarchical step of the proposed system which we describe next.

B. THE HIERARCHICAL STEP

1) FINDING COUNTRY LEVEL FOCUS OF THE DOCUMENT

In this step, we determine the country level focus/foci of the document under consideration. The list of locations in the web page content is in list named L. The data structures used by the algorithm are illustrated below. For each entry in L, a hashmap LM is constructed in lines 9 to 12 of Algorithm 1. An entry of LM is a tuple of the form, $< L_i, < fval, tval >>$, where L_i is the location name, $fval$ is the frequency of the location, and $tval$ is the tag value of the location. The function $findTagValue$ on line 12 finds the tag value of a place name (say L_i) under consideration in the web page W. For this, the position of the place name in the web page is considered. We give the highest weight (value 4) if the place name occurs in the title, a weight of value 2 if it occurs in the header tags and a low weight of value 1 if it occurs in the head tags. The intuition behind this is that if a place name occurs in the title of the document, it is very likely that the document is about that particular place name. If it occurs in the sub-headings, we give it less importance than if it had occurred in the title but more than the case if it had occurred as normal text. All the entries in the gazetteer for each location L_i in LM are obtained and stored in ML_i. ML_i is a list where each entry is a 4-tuple $< lat, long, pval, c >$ corresponding to the latitude, longitude, population and the country of the entry. ML is a table of such lists. This data structure gets populated in lines 15 and 16 of Algorithm 1. A set CL_i consisting of distinct countries in ML_i is constructed. With each country, a population value is associated using the gazetteer. This is done in lines 17 and 18 of Algorithm 1. Now the hashmap CM is constructed, in which, each entry is of the form $< c, < fval, tval, pval, w >>$ where the country c is the key and the 4-tuple $< fval, tval, pval, w >$ is the value found earlier. This is done in lines 20 to 29 of the Algorithm 1.

In order to determine the country level focus/foci, we consider three parameters. First, the frequency of occurrence of the place name in the document is taken into account since a higher frequency of a place name indicates its importance in the document. The second parameter is the tag value. The last parameter considered is the population of the place. Higher population is indicative of the importance of the place. We have given weights to each criterion, w_1 for frequency, w_2 for tag value and w_3 for population value, where $w_1 > w_2 > w_3$ (typically $w_1 = 0.6, w_2 = 0.3, w_3 = 0.1$). These values have been obtained empirically after thorough experimentation. We have considered normalised values of the parameters and then a weighted sum is computed for each country.

An entry in the gazetteer corresponding to a place name or location l is called as a match m for l. A location is said to belong to a country c if at least one of its matches belongs to c. We find the country with the maximum weight and mark the value of the maximum weight as W_max. If the difference between the weight assigned to a country c and W_max is less than a threshold θ, c is included in the country focus list. θ has been determined empirically by thorough experimentation. The Algorithm 1 finally returns the country focus list CFL and the constructed hashmap LM.

2) FINDING THE CONTINENT LEVEL FOCUS

The pseudo code of the hierarchical step is given in Algorithm 1 and its returns the country level focus list CFL along with the hashmap LM.

As the country names in the world are unique, we can take the continent corresponding to each country from the country level focus list to determine the continent level focus list for a web page. This is explained in lines 5 to 7 of Algorithm 2.

After finding the country and continent level foci, the further steps of the algorithm are invoked for the documents which have more than one places present in their text. For web pages having only one place in the content, we find all the entries of that place name from the gazetteer. The focus reported for the document is the $< latitude, longitude >$ of the entry having the highest population among all the entries for that place name. This is based on the observation that, in most cases, a place with the highest population is likely to be the focus with a high probability.

For the documents with multiple place names, we proceed as follows. First the output of hierarchical step is given as input to the algorithm for pruning points given in Algorithm 3. If there are entries in LM which do not belong to any country in the country focus list CFL, then such entries are removed. If required, the particular location is also removed. The modified LM is now ready to be given as input to the geometrical step.

C. THE GEOMETRICAL STEP

The geometrical step involves the following operations.

1. Formation of the bounding rectangle as per the function named $BRFA$.

2. Iteratively pruning of points from the rectangle as per Algorithm 4.

In this algorithm, in each iteration points from the rectangle are removed and $BRFA$ function is called.

In order to form the bounding rectangle of the points obtained after Algorithm 3, we place the points on the map of the world. We consider the x-coordinate as the longitude of a point and the y-coordinate as its latitude. In order to do this, we consider the earth as a flat surface. The proposed algorithm considers all the entries in LM together and forms a list BRL containing all the points. The function $BRFA$ determines a bounding rectangle, which encloses all these points such that they are either inside the rectangle or on its perimeter. Note that for a set of points on a two-dimensional surface, a rectangle is the figure with the minimum area that encompasses those points. The algorithm for determination for bounding rectangle for a given set of points is

Data Structures used in Algorithm 1 *The Hierarchical Step Algorithm*
Location Map = LM = Hashmap < Location, < fval, tval >>
Country Map = CM = Hashmap < Country, < fval, tval, pval, w >>
Matches of Location = ML = a table of lists.

as given in section III-C.1. Once the bounding rectangle is formed, the algorithm proceeds in an iterative fashion. In each iteration using some heuristics, we consider eliminating a point from this bounding rectangle. Algorithm 4 mentions the details.

1) CONSTRUCTION OF THE BOUNDING RECTANGLE

The algorithm places all points in consideration in a 2-D area as stated earlier. We consider the earth as a flat surface and place the points according to their x and y co-ordinates. As earth is actually elliptical/ spherical in shape, 180^0 longitude = $-\ 180^0$ longitude. Hence, the points having longitude values as 180^0 and $-\ 179^0$ are near to each other in reality. However, with the flat surface assumption, they will be considered as far from each other, as shown in figure 2. This has been appropriately taken in to account by the proposed function *BRFA* for the computation of the bounding rectangle. This is explained next.

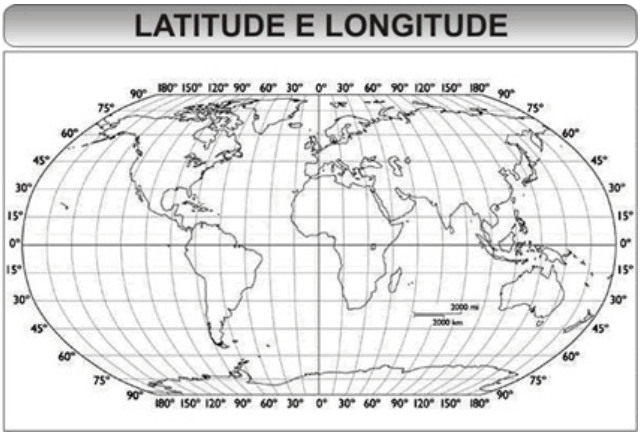

FIGURE 2. Flat surface view of earth with latitudes and longitudes.

We sort all the points under consideration in the non-descending order of their longitude values. We consider the consecutive points in the sorted order of the longitude values and calculate the difference in their longitude values, in only one direction (say clockwise). If the difference is less than or equal to 0, 360 is added to the difference. The pair of points having minimum difference in their longitude values will be our right and left vertical edges of the bounding rectangle. Going clockwise, the longitude value of the point which is encountered first makes the right vertical edge and the longitude of the point which is encountered immediately next forms the left vertical edge of the bounding rectangle. The maximum and minimum latitude values amongst the points under consideration will be the upper and lower horizontal edges respectively of the bounding rectangle.

For example, consider that we have five points $x1$ to $x5$ with their <latitude, longitude> values as follows.

x1<30, −120>, x2<−12, 130>, x3<19, −160>, x4<−50, 160>, x5<42, 180>

After sorting in non-descending order of longitudes, the points are as follows:

x3<19, −160>, x1<30, −120>, x2<−12, 130>, x4<−50, 160>, x5<42, 180>

The difference between consecutive longitudes in the sorted list is calculated as follows.

$$-160-(-120) = -160+120 = -40 \quad (1)$$
$$-120-(130) = -250+360 = 110 \quad (2)$$
$$130-(160) = -30+360 = 330 \quad (3)$$
$$160-(180) = 20+360 = 340 \quad (4)$$
$$180-(-160) = 340 \quad (5)$$

The pair with the minimum difference consists of points $x1$ and $x2$. These form the right and left vertical edges of bounding rectangle respectively. −50 will form the lower horizontal edge of the bounding rectangle since that is the least latitude value among the set of points. Correspondingly, 42 will form the upper horizontal edge. Thus we obtain the bounding rectangle of minimum area encompassing all the points with the co-ordinates (42, 130) and (−50,−120) of its left upper and right bottom corner respectively. For the above example the bounding rectangle formed is as shown in the figure 3 and 4.

2) PRUNING OF POINTS IN GEOMETRICAL STEP

After the bounding rectangle is formed, we find dV, the distance between the vertical edges of the rectangle and dH, the distance between the horizontal edges. The number of points on the left and the right vertical edges are indicated by VL and VR respectively. Similarly, HU and HL indicate the number of points on the upper and lower horizontal edges respectively. If dV is greater than dH, we eliminate points from either the left or the right vertical edge whichever has less number of points on it. If dH is greater than dV, we eliminate points on either the upper horizontal or the lower horizontal edge whichever has less number of points on it. If dV and dH are equal, we find the centroid first and eliminate points from the edges which are farthest from the centroid. The reason for eliminating the points from the edge with less number of points is that it is more likely that the edge with more number of points is close to the focus or it

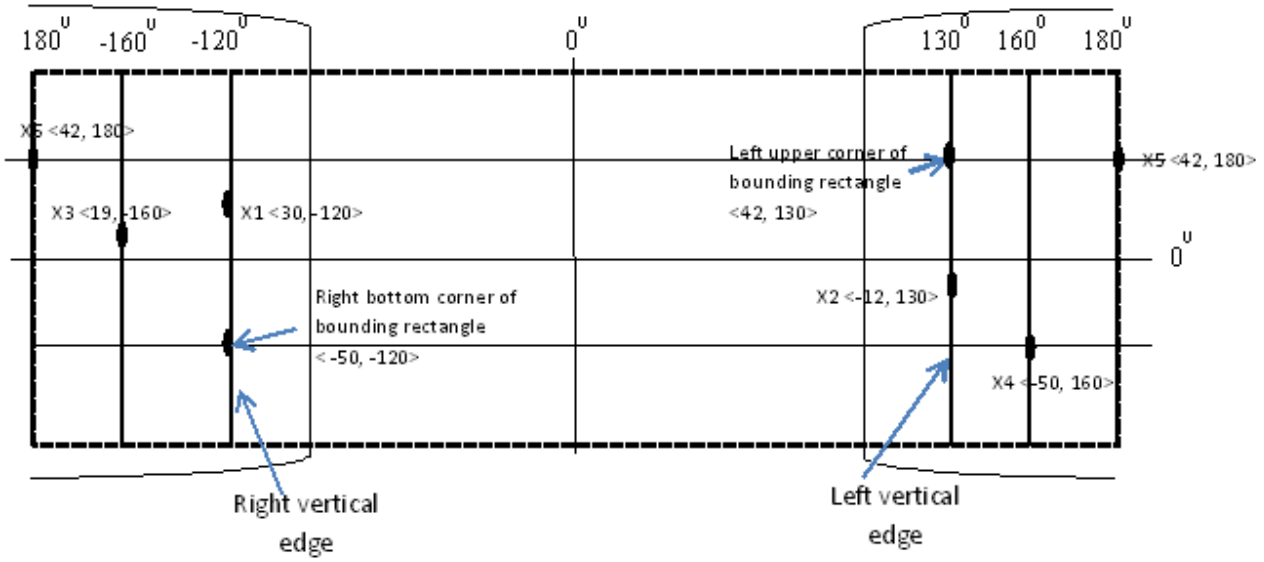

FIGURE 3. Formation of bounding rectangle on flat surface of earth.

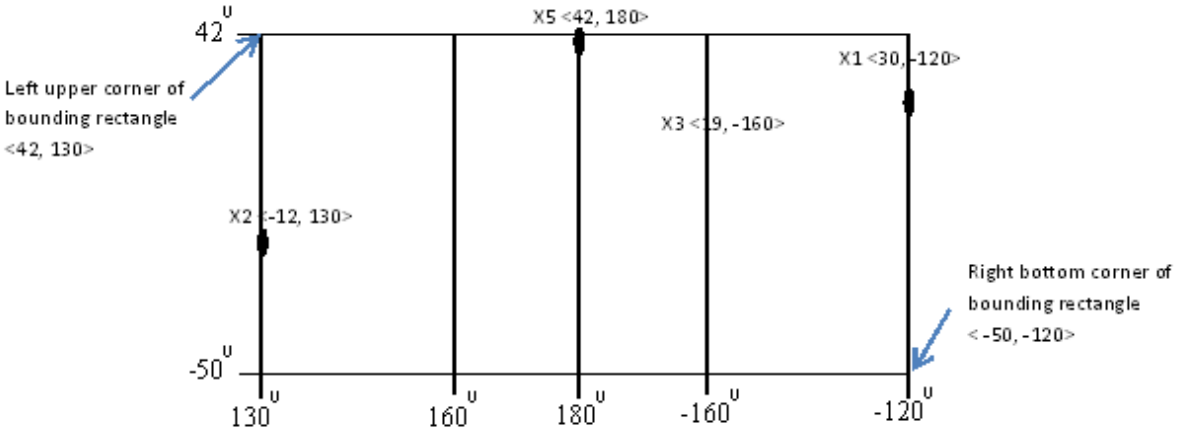

FIGURE 4. Bounding rectangle for given example.

may even contain the focus. Algorithm 4 gives the details. The intuition behind removal of the points is to attempt to reach to the smallest area in which all the place names in the document get a unique latitude and longitude value. It is based on the assumption that if a document mentions several place names, it is quite likely that the places are all close by. A striking feature of the geometrical step is that the disambiguation and focus finding is performed together by pruning the points. The pruning is performed until we are left with at least one match for 80% of the locations. The parameter of 80% is decided empirically after rigorous experimentation.

The iterative algorithm of the geometrical step is explained with the help of following example. Consider a document with five unique geographical locations. The matches of location 1 obtained from the gazetteer are indicated by circles. Those for location 2 are indicated by diamonds, location 3 by squares, location 4 by stars and location 5 by triangles. The figure 5 shows the legends used.

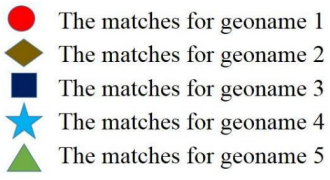

FIGURE 5. Legend used for the geometrical step example.

The initial stage where all the points of the all the locations are placed on a 2-D area is shown in figure 6.

In the first iteration, the distance between the vertical edges of the bounding rectangle is more than that between the horizontal edges. So, we remove the points on the vertical edge. Since one point is there on the left edge and three points on the right edge, we remove the point on the left edge and reduce the size of the bounding rectangle. In a similar manner, points are removed in iterations 2, 3, 4,

Algorithm 1 The Hierarchical Step

Prerequisite: Locations set L extracted by Achemy API from webpage
Input:
L = Set of n distinct locations in the web page content
W = the web page itself
G = GeoNames Gazetteer
Output: Country level focus list CFL, Hashmap LM
begin
LM ← <>
for *each location* L_i *in input set L* **do**
    insert an entry in LM with key as L_i
    LM.L_i.fval ← frequency of L_i in W
    LM.L_i.tval ← findTagValue(L_i, W)
CM ← <>
foreach *location* L_i *in Location Map LM* **do**
    ML_i = <>
    foreach *entry X of* L_i *in Gazetteer G* **do**
        insert X in ML_i
    CL_i ← select distinct X.country from ML_i
    PL_i ← select max(X.pval) from ML_i group by CL_i
    foreach *country Y in* CL_i **do**
        if $Y \notin CM$ **then**
            Insert an entry in CM with Y as key
            CM.Y.tval ← L_i.*tval*
            CM.Y.fval ← L_i.*fval*
            CM.Y.pval ← Y.pval
        else
            CM.Y.tval ← CM.Y.tval + L_i.*tval*
            CM.Y.fval ← CM.Y.fval + L_i.*fval*
            CM.Y.pval ← max($Y.PL_i$, CM.Y.pval)
    <minfval, maxfval> ← findMinmaxfval(CM)
    <mintval, maxtval> ← findMinmaxtval(CM)
    <minpval, maxpval> ← findMinmaxpval(CM)
foreach *country z in CM* **do**
    z.fvaln ←(z.fval − minfval)/(maxfval − minfval)
    z.tvaln ←(z.tval − mintval)/(maxtval − mintval)
    z.pvaln ←(z.pval − minpval)/(maxpval − minpval)
    z.w ← w_1*z.fvaln+w_2*z.tvaln+w_3*z.pvaln
W_{max} ← findMaxWeight(CM)
foreach *country Q in CM* **do**
    if $(W_{max} - Q.w) <= \theta$ **then**
        insert Q in CFL
return(CFL, LM)
end

and 5. In iteration 6, the distance between horizontal edges is more. So, we have to remove a point on the horizontal edge. However, there is a tie since both the upper and lower horizontal edges contain one point each. Hence, we calculate

Algorithm 2 Algorithm to Find Continent Level Focus List

Prerequisite: HierarchicalStepAlgo has returned CFL, LM
Input: CFL, Gazetteer G
Output: Continent level focus list COFL
begin
foreach *country C in CFL* **do**
    Find the continent CO to which C belongs
    insert CO in COFL
return(COFL)
end

Algorithm 3 Algorithm to Prune Points After Hierarchical Step

Prerequisite: The hierarchical step has returned CFL, LM
Input: List CFL, Hashmap LM
Output: Modified LM
begin
foreach *location* L_i *in LM* **do**
    foreach *entry of M in* ML_i **do**
        if *M does not belong to a country present in CFL* **then**
            Remove M from LM
            if *M was the only entry in* L_i **then**
                Remove L_i from LM
return(LM)
end

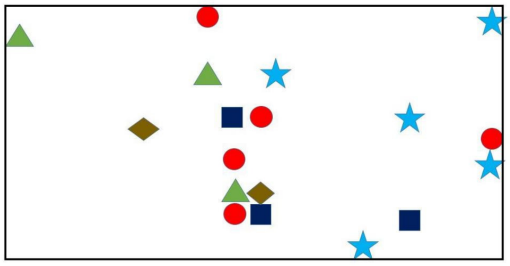

FIGURE 6. Initial stage when the geometrical step of the algorithm begins.

the centroid of all the points inside the bounding rectangle. We consider the distance of each point on both the upper and lower horizontal edges from the centroid. As the point on the upper horizontal edge is farthest, we remove that point. Similarly, iterations 7,8,9 are performed. Now, we have 4 locations remaining in *LM*. This is 80% of 5 which was the initial count of locations in *LM*. Hence in iteration 10, no further points can be removed. So the geometrical step of the algorithm stops here. The output after iteration 10 is shown in the figure 7.

```
1  Function name: BRFA (Bounding rectangle Formation
   Algorithm)
2  Prerequisite: List named BRL contains all points to be
   considered for formation of bounding rectangle
3  Input: List BRL of size n
4  Output: Co-ordinates of left upper (LU) and right lower
   (RL) vertices of the bounding rectangle
5  begin
6  Sort all points in BRL in non-descending order of their
   longitude values
7  diffmin ← 361
8  i ← 0
9  while i < n do
10     j ← (i+1)mod n
11     P_i ← BRL[i]
12     P_j ← BRL[j]
13     diff ← LongitudeOf(P_i) − LongitudeOf(P_j)
14     if diff <= 0 then
15         diff ← diff + 360
16     if diffmin > diff then
17         diffmin ← diff
18         longLU ← LongitudeOf(P_j)
19         longRL ← LongitudeOf(P_i)
20     i ← i+1
21 latLU ← findPointMaxLat(BRL)
22 latRL ← findPointMinLat(BRL)
23 return(latLU, longLU, latRL, longRL)
24 end
```

D. FOCUS REPORTING STEP

Algorithm 5 is invoked for finding the final focus of the document after the pruning of points by the geometrical step has been performed. In this step, we report almost two foci for a page since it is found that most of the pages have either one or two foci. If the number of points in *LM* after execution of Algorithm 4 is less then the threshold value, we report the single entry as focus. If threshold number of points are present in *LM*, then we check if one of them subsumes the other. If so, we report the contained or subsumed region as the focus. A place or a region x is said to be contained by another region y if the administrative boundaries of a region y subsume the administrative boundaries of region x. For example, New York City is contained in New York State. This information can be suitably extracted from the gazetteer. If there is no such hierarchical relationship between the two points, then we report both as the foci. If in case the number of points in *LM* is greater than the threshold, we check if any of the points represents a large region like a country or a continent. If so, and if any of its contained region is also in *LM*, we remove the large region from *LM*. We then compute the centroid of all the points. We then find the two points closest to the centroid and report them as the foci.

Algorithm 4 Pruning of Points in the Geometrical Step

```
1  Pre-requisite: Algorithm for hierarchical step, finding
   continent level focus and the algorithm for pruning
   points has been executed
2  Input: Hashmap LM, Array of lists ML
3  Return type: Modified Hashmap LM, ML
4  begin
5  s ← |LM|
6  t ← min(s, ⌈s ∗ 0.8⌉ + 1)
7  LNTR ← <>
8  while s > t do
9      foreach Location L_i in LM do
10         foreach entry M of L_i in ML_i do
11             insert (M.lat, M.long) in BRL
12     <latLU, longLU, latRL, longRL> ←
13                                 BRFA(BRL, LNTR)
14     dV ← abs(latLU - LatRL)
15     dH ← abs(LongLU - LongRL)
16     VL ← findnumberOfPoints(longLU)
17     VR ← findnumberOfPoints(LongRL)
18     HL ← findnumberOfPoints(latRL)
19     HU ← findnumberOfPoints(latLU)
20     if dV > dH then
21         if VR > VL then
22             foreach point P on left vertical edge do
23                 RemovePoint(P, LM, L_i, LNTR, t)
24         else if VR < VL then
25             foreach point P on right vertical edge do
26                 RemovePoint(P, LM, L_i, LNTR, t)
27         else
28             CTR ← calcCentroid(LM, LU, RL)
29             foreach point Q in LM do
30                 distQ ← calcDistance(Q, CTR)
31                 insert (Q, distQ) in PM
32             maxD ← calcMaxDistance(PM)
33             foreach point P in PM do
34                 if distP = maxD then
35                     LM ← RemovePoint
36                           (P, LM, L_i, LNTR, t)
37     else if dV < dH then
38         same as that for dV > dH considering horizontal
           edges instead of vertical edges. All the steps are
           symmetrical.
39     s ← |LM|
40 end
```

E. TIME COMPLEXITY OF THE ALGORITHM

The algorithm's time complexity is dominated by the second step. If we denote m as the number of place names identified by Alchemy API in a particular document and n as the number of references of that place name obtained from the gazetteer, then the complexity of the algorithm can be denoted

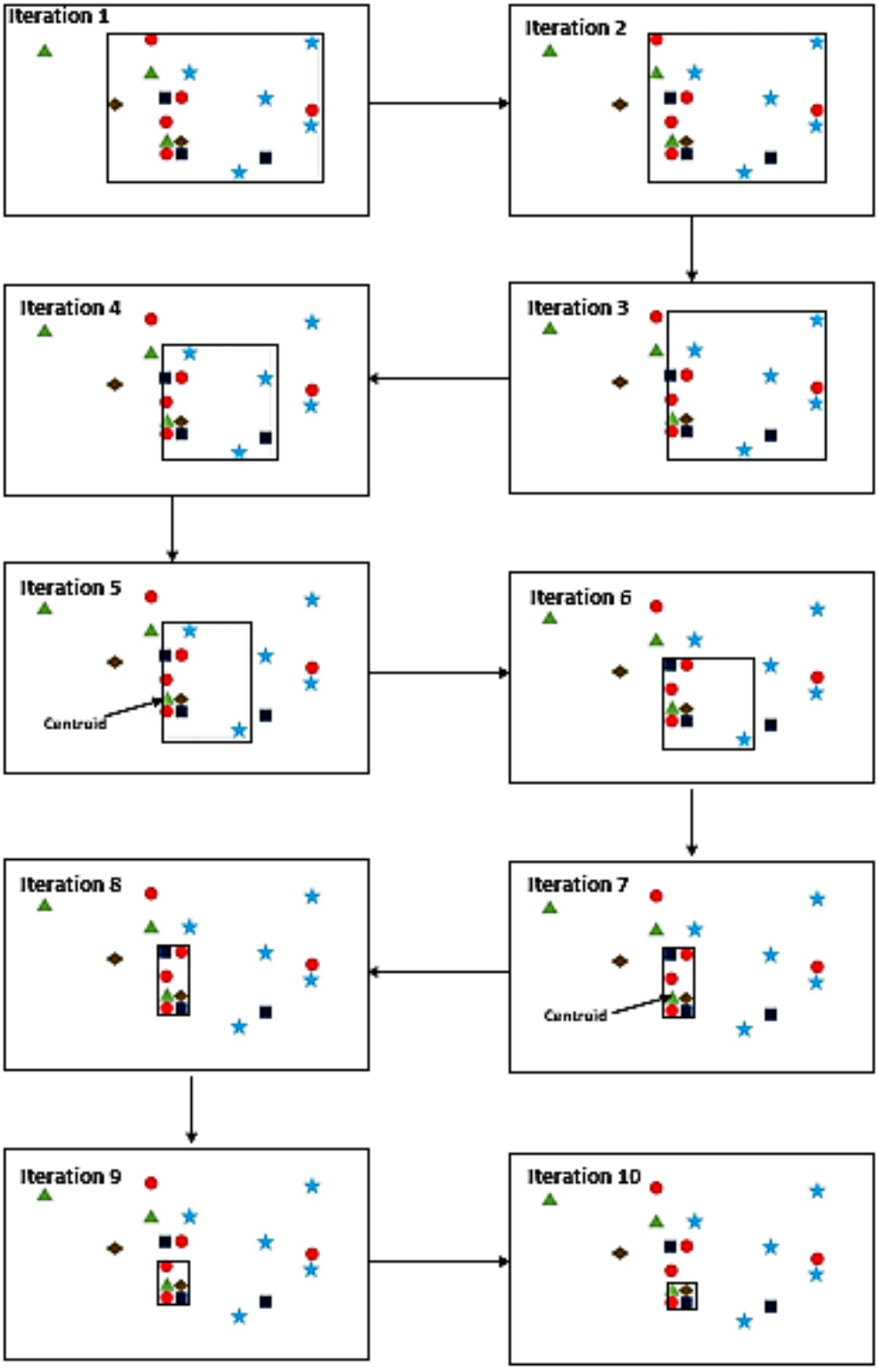

FIGURE 7. Iterations of the geometrical step algorithm.

as O(m*n). This is because step 2 considers each reference out of a total of m*n references and in each iteration at least one point/reference is pruned.

IV. EXPERIMENTS AND RESULTS

For our experimental analysis, we have used the following datasets.

```
Function Name: RemovePoint
Note: Li is the Location to which Point P belongs.
LNTR is the list of locations not to be removed
Input: Point P<lat, long>, Hashmap LM, Location L_i,
List LNTR, threshold value t
Output: List LNTR, Hashmap LM
begin
s ← |LM|
if (s > t) and (L_i ∉ LNTR) then
    remove P from LM
    if P was last point belonging to L_i then
        remove L_i from LM
else
    if L_i ∉ LNTR then
        insert L_i in LNTR
return <LNTR, LM>
end
```

1) The Wikipedia dataset - which is a manually annotated subset of the Wikipedia pages. This dataset is available online[1] and it contains 424,171 articles whose geographical focus is reported as a tuple $< latitude, longitude >$.
2) The Open Directory Project (ODP) [3] - in which 940,468 links are available in a hierarchical form. For example, a link for the *New York City* will be placed in a hierarchy as *North America, USA, New York, New York City*.

A. EXPERIMENTS AND RESULTS ON THE WIKIPEDIA DATASET

In pre-processing, the pages from the dataset are input to Alchemy. Some of the pages are filtered out based on the place names retrieved by Alchemy. We do not consider the pages for which no place names are retrieved. We also do not consider pages which might have place names, however, none of the place names are present in the gazetteer. In Table 1, the first column shows the filtering step performed and the second gives the count of pages remaining after the filtering step.

The Wikipedia dataset has pages along with their actual focus as a tuple $< latitude, longitude >$. Additionally, we need to annotate each page with its country and continent. In order to find these, we use the following technique. If $< latitude, longitude >$ of the actual focus is present in the gazetteer, we find the continent and country from the gazetteer. If it is not present, we find all those places from the gazetteer that differ by almost $+0.05$ in *both* the latitude and longitude values. The country and continent of the majority of those entries is considered as the country level focus and the continent level focus respectively for the page.

[1]https://fusiontables.google.com/DataSource?dsrcid=423292

Algorithm 5 The Focus Reporting Step

```
Pre-requisite: The geometrical step has been executed
Input: Hashmap LM
Output: Focus List FL
begin
if noOfPoints <= θ' then
    if noOfPoints = θ' then
        if P subsumes P' then
            insert P' in FL
        else
            if P' subsumes P then
                insert P in FL
            else
                insert P in FL
                insert P' in FL
    else
        insert P in FL
else
    foreach P ϵ LM do
        foreach P' ϵ LM and P' ≠ P do
            if (P subsumes P') and (P is a country or a
            continent) then
                remove P from LM
            else
                if (P' subsumes P) and (P' is a
                country or a continent) then
                    remove P' from LM
    if noofPoints in LM <= θ' then
        foreach Point P ϵ LM do
            insert P in FL
    else
        CTR ← centroid(LM)
        Two points nearest to CTR in LM are inserted in
        FL
return(FL)
end
```

After applying the proposed approach, the continent level focus of 97.02% articles and the country level focus of 95.57% articles are correctly reported.

We also find out the error in kilometres (km) as the distance of the reported focus from the actual focus of a document. As the earth is elliptical in shape, we used the haversine distance formula [2] to find distance between two points. The following procedure is adopted to calculate the error. As stated earlier, the proposed approach returns either one focus as a tuple $< latitude, longitude >$, or two foci (with two tuples) for each web page. For the

TABLE 1. Pre-processing of the Wikipedia Dataset.

Filtering step	Pages Remaining after filtering
Initially	424,171
After removal of pages parsed by Alchemy which have no place name in their text	392,783
After removal of pages which have some place names parsed by Alchemy from their text but none of those place name entries are found in the gazetteer	386,229

TABLE 2. Distribution of error in km for the Wikipedia dataset.

Error in k)	% of web pages	Cumulative % of web pages
< 5	43.04	43.04
< 10	10.18	53.22
< 15	07.15	60.37
< 30	11.49	71.86
< 50	04.79	76.65
< 100	05.20	81.85
< 300	07.43	89.28
< 500	02.30	91.58
< 1000	02.12	93.71

Wikipedia dataset, the actual focus is available as a single tuple $< \ latitude, longitude \ >$. If the proposed approach reports one focus, we report the error as the distance between the reported focus and the actual focus given with the webpage. If the algorithm reports two foci, we find the distance of both the reported foci from the actual focus. We consider the minimum of the two distances as the error.

Table 2 gives the distribution of error in km for the Wikipedia dataset.

As evident from the table, it is found that more than 75% of the web pages have an error less than 50 km. Moreover, for more than 60% of the documents, the error in the distance between the actual focus and the reported focus is within 15 km i.e. less than 10 miles for the Wikipedia dataset. Thus we can say that the proposed approach is quite accurate and is able to give a correct geotag to a webpage with a granularity up to a city or a locality within a city. The median error is 8.17 km, which is lower than that reported by Melo and Martins [21]. The lowest median error of 4.2 km as shown in the table 6 has been reported in the work by Laere et al. [14]. However, they have considered a small subset of the Wikipedia dataset, called the UK dataset. Focussed and accurate gazetteers (such as OpenNames [5]) are available for the UK region. Since, the dataset that we use consists of all the places in the world, such focussed and accurate gazetteers are not available. The gazetteer used in this work is the GeoNames gazetteer, which we obtained in January, 2015. Each entry in GeoNames consists of name of the location, its latitude, longitude, country to which it belongs, population and other miscellaneous information. In that, it was observed that the population value was not properly given in many entries. This has resulted in error in the proposed approach. The accuracy of our system would increase if updated data is used.

We have done analysis of the error distribution with respect to the number of place names in the document. The details are in the table 3. There is no pattern observed in the error distribution with respect to the number of place names present in the document and hence the technique is not sensitive to this variable.

The haversine distance, d (in km), between two points (lat1, long1) and (lat2, long2) is calculated as given by the equations 6 to 10

$$\Delta lat = lat_2 - lat_1 \tag{6}$$

$$\Delta long = long_2 - long_1 \tag{7}$$

$$a = sin^2\langle \Delta lat \div 2\rangle + cos\langle lat_1\rangle \cdot cos\langle lat_2\rangle . sin^2\langle \Delta long \div 2\rangle \tag{8}$$

$$c = 2 * arctan\langle \sqrt{a} \div \sqrt{\langle 1 - a\rangle}\rangle \tag{9}$$

$$d = R * c \tag{10}$$

R is the radius of the earth = 6371 km

d is the distance between the points $(lat_1, long_1)$ and $(lat_2, long_2)$ in km

TABLE 3. Error distribution with respect to the number of place names in a document (Wikipedia).

Number of place names	1	2	3	4	5	6	7	8	9	10	>10
Number of documents	29335	21832	41000	46897	52251	36019	29494	20874	16606	13454	78447
distribution of error in km	Cumulative percentage of documents										
<= 5	n/a	41.04	41.78	41.66	33.7	41.13	41.57	48.5	51.49	51.66	48
<= 10	n/a	52.35	53.46	51.88	44.93	51.93	51.5	58	60.95	61.13	56.64
<= 15	n/a	58.15	60.5	59.46	56.28	60.79	59.04	63.62	66.04	66.57	61.35
<= 30	n/a	67.27	71.91	70.82	75.34	74.61	71.86	72.74	73.8	74.09	69.12
< =50	n/a	71.72	77.15	76.3	80.46	78.93	76.9	76.7	78.07	78.07	73.73
<= 100	n/a	76.52	82	82.03	85.11	84.61	82.65	81.48	82.52	82.64	79.21
<= 300	n/a	84.46	90.17	89.84	91.12	91.06	89.76	88.91	89.32	89.58	87.62
< =500	n/a	87.77	92.58	91.8	92.88	93.14	91.84	91.58	91.65	91.81	90.26
<= 1000	n/a	90.75	94.68	93.89	94.37	95.01	93.82	93.63	93.88	93.91	92.77

TABLE 4. Comparison with existing approaches.

Approach	Corpus	Median
Proposed Approach	Wikipedia dataset	8.17 km
Melo and Martins [20]	Wikipedia dataset	8.9 km
Laere et al. [14]	Wikipedia (UK dataset)	4.2 km

TABLE 5. Pre-processing of the ODP Dataset.

Filtering step	Pages Remaining after filtering
Initially	940,468
After removal of pages parsed by Alchemy which have no place name in their text	627,661
After removal of pages which have some place names parsed by Alchemy from their text but none of those place name entries are found in the gazetteer	595,172

B. EXPERIMENTS AND RESULTS ON THE ODP DATASET

The ODP dataset is pre-processed using the same filtering steps as those performed on the Wikipedia dataset. The number of pages filtered out are shown in the table 5.

We apply the proposed approach on around 0.6 million ODP web pages left after the filtration steps. In the ODP dataset, the web documents are arranged in a hierarchical way in the regional directory [4]. Some examples of web pages showing how the web pages are stored in the ODP dataset are given below.

As per the above structure shown in 6, the links depict a hierarchy. The first level of the hierarchy is always a continent and the second level is always a country. However, after these two levels, different entries have different names for the lower levels. As shown above, after the first two levels, some links may have names of levels as ''(State/County/Location)''. Some other links have only one lower level with the name ''(City)''. The nomenclature of the levels is different for different regions in the world depending on the administrative divisions of different countries. For a particular region the third level could be a state, while for some other region it could be called as county.

Our algorithm reports the focus as a tuple $<$ *latitude*, *longitude* $>$. We need to find up to what level the reported focus is correct as per the hierarchy present in the ODP dataset. For this, we query the reported focus in the gazetteer and find the location corresponding to that value of $<$ *latitude*, *longitude* $>$. We also find the administrative hierarchy of the location as reported in the gazetteer. We call this as the reported hierarchy. Starting from the administrative level of the location in the reported hierarchy, we match the calculated hierarchy and the actual hierarchy of the ODP link. If all the entries match, then we say that the reported focus is correct up to the level of the location.

For example, for a web page the actual hierarchy in the ODP dataset is as follows:

NorthAmerica/*UnitedStates*/*Texas*/*Dallas*/*WhiteRockLake*.

For the above web page, the algorithm reports the focus as (32.8341, −96.72282). For the reported focus, we find out the reported hierarchy from the gazetteer. The reported hierarchy for the above page is:

NorthAmerica/*UnitedStates*/*Texas*/*Dallas*/*WhiteRockLake*.

We search the location ''(WhiteRockLake)'' in the actual hierarchy. It matches at level 5. Hence, we compare the names of all the higher levels in both the actual and reported hierarchy. Since they match, we report that the focus is correct up to level 5. In case, the entry of ''(White Rock Lake)'' would not have been there, then we would have checked for ''(Dallas)'' in the actual hierarchy. If that too would have not been there, then we would have searched for ''(Texas)'' and so on. We thus report the level till which the focus is correctly calculated by the proposed algorithm.

Table 4 shows the percentage of links correctly classified till continent level (level 1), country level (level 2), and then level 3, level 4 and so on.

The proposed approach reports the correct continent level focus for 93.07% of the web pages and the correct country level focus for 90.27% of them. For almost 80% of the pages, the focus is correctly reported up to level 3.

C. COMPARISON WITH THE EXISTING TECHNIQUES

We have compared the results of proposed algorithm with those of the graph ranking algorithm [19] as shown in table 8. The dataset (ODP) used is same in both the cases and we can see that the proposed algorithm performs better. The web-a-where system performs slightly better than our approach. However the size of the dataset considered in that system is *30 times* smaller than the size used by us.

TABLE 6. Pre-processing of the ODP Dataset.

1)$Top/Regional/SouthAmerica/Colombia/Health$ which is having hierarchy $Continent/Country$ i.e. $level1/level2$
2)$Top/Regional/NorthAmerica/UnitedStates/Texas/Localities/D/Dallas/TravelandTourism/Parks/WhiteRockLake$ which is having hierarchy $Continent/Country/State/County/Location$ i.e. $level1/level2/level3/level4/level5$
3)$Top/Regional/SouthAmerica/Colombia/Localities/Cali$ which is having hierarchy $Continent/Country/City$ i.e. $level1/level2/level3$
4)$Top/Regional/NorthAmerica/ArtsAndEntertainment/Libraries$ having hierarchy as only $Continent$ i.e. $level1$

TABLE 7. Levels of Hierarchy of ODP dataset.

Name of the level	Pages with focus reported correctly up to a particular level
Continent level (Level 1)	93.07%
Country level (Level 2)	90.27%
Level 3	79.54%
Level 4	50.11%
Level 5	31.41%
Level 6	20.51%
Level 7	0.00%

TABLE 8. Comparison with existing approaches.

Approach	Corpus	Sample Size	Continent-Level Correct Focus	Country-Level Correct Focus
Proposed Approach	Wikipedia articles	400,000	97.02%	95.57%
Proposed Approach	Open Directory Project	600,000	93.07%	90.27%
Web-a-Where [7]	Open Directory Project	20,000	96%	92%
Graph Ranking Algorithm [19]	Open Directory Project	1,000,000	92%	85%

V. CONCLUSION AND FUTURE WORK

In this paper we have presented an algorithm for geotagging of the web documents. It does not disambiguate individually the place names occurring in the document content and considers all the place names together. The algorithm consists of three steps - namely, the hierarchical step, the geometrical step and the focus reporting step. The algorithm is tested on two datasets namely, the Wikipedia and the ODP datasets. We have achieved correct continent level focus in 97.02% and correct country level focus in 95.57% of the cases for the Wikipedia dataset. Moreover, for more than 60% of the documents, the error in the distance between the actual focus and the reported focus is within 15 km i.e. less than 10 miles for the Wikipedia dataset. The median error for the Wikipedia dataset is 8.17 km. Similarly, we report the continent level focus correctly in 93.07% and country level focus correctly in 90.27% of the cases for the ODP dataset.

The proposed approach can be used in the implementation of a complete Geographical Information Retrieval (GIR) system and can facilitate spatial indexing. The approach is simple, intuitive and performs better than the existing approaches.

This work has several future directions.

In the proposed approach, we have not considered a document if it does not have any placename. As pointed out by Melo and Martins [21], this can be done by using a classifier that classifies a document into one of the two classes - one which have a geotag and another which do not have any spatial information mentioned in them.

Though we have not considered Twitter data, the proposed technique can be applied to twitter data and other social media data with some customised rule based mining methods to come up with hybrid approaches. The method can also be used for other datasets like news articles corpus, historical documents etc. It can be also be combined with supervised learning methods to create hybrid approaches.

REFERENCES

[1] *The Alchemy API*. Accessed: Sep. 2015. [Online]. Available: https://www.ibm.com/watson/alchemy-api.html
[2] *The Haversine Formula for Distance Calculation*. Accessed: Jul. 2015. [Online]. Available: http://www.ig.utexas.edu/outreach/googleearth/latlong.html
[3] *The Open Directory Project Database*. Accessed: Dec. 2015. [Online]. Available: http://www.dmoz.org/
[4] *The Open Directory Project database—Regional Directory*. Accessed: Dec. 2015. [Online]. Available: http://www.dmoz.org/Regional/
[5] *The OS Open Names U.K. Gazetteer*. [Online]. Available: http://ordnancesurvey.co.uk/business-and-government/products/os-open-names.html
[6] E. Amitay, N. Har'El, R. Sivan, and A. Soffer, "Finding the geographic focus of Web-pages," in *Proc. Workshop Geograph. Inf. Retr. (SIGIR)*, 2004, pp. 1–2.
[7] E. Amitay, N. Har'El, R. Sivan, and A. Soffer, "Web-a-where: Geotagging Web content," in *Proc. 27th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2004, pp. 273–280.
[8] G. Andogah, G. Bouma, and J. Nerbonne, "Every document has a geographical scope," *Data Knowl. Eng.*, vols. 81–82, pp. 1–20, Nov./Dec. 2012.
[9] M. Chen, X. Lin, Y. Zhang, X. Wang, and H. Yu, "Assigning geographical focus to documents," in *Proc. 18th Int. Conf. Geoinformatics*, Jun. 2010, pp. 1–6.
[10] C. D'Ignazio, R. Bhargava, E. Zuckerman, and L. Beck, "CLIFF-CLAVIN: Determining geographic focus for news," in *Proc. NewsKDD, Data Sci. News Publishing (KDD)*, 2014, pp. 1–5.
[11] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *Proc. 43rd Annu. Meeting Assoc. Comput. Linguistics*, 2005, pp. 363–370.
[12] J. M. Johns, J. Rounds, and M. J. Henry. (2017). "Multi-modal geolocation estimation using deep neural networks." [Online]. Available: https://arxiv.org/abs/1712.09458
[13] G. Kordopatis-Zilos, S. Papadopoulos, and I. Kompatsiaris, "Geotagging text content with language models and feature mining," *Proc. IEEE*, vol. 105, no. 10, pp. 1971–1986, Oct. 2017.
[14] O. Van Laere, S. Schockaert, V. Tanasescu, B. Dhoedt, and C. B. Jones, "Georeferencing wikipedia documents using data from social media sources," *ACM Trans. Inf. Syst.*, vol. 32, no. 3, 2014, Art. no. 12.
[15] J. L. Leidner, G. Sinclair, and B. Webber, "Grounding spatial named entities for information extraction and question answering," in *Proc. HLT-NAACL Workshop Anal. Geograph. References*, vol. 1, 2003, pp. 31–38.
[16] M. D. Lieberman, H. Samet, and J. Sankaranarayanan, "Geotagging with local lexicons to build indexes for textually-specified spatial data," in *Proc. IEEE 26th Int. Conf. Data Eng. (ICDE)*, Mar. 2010, pp. 201–212.
[17] M. D. Lieberman, H. Samet, and J. Sankaranayananan, "Geotagging: Using proximity, sibling, and prominence clues to understand comma groups," in *Proc. 6th Workshop Geograph. Inf. Retr.*, 2010, Art. no. 6.
[18] B. Martins, M. Chaves, and M. J. Silva, "Assigning geographical scopes to Web pages," in *Proc. Eur. Conf. Inf. Retr.* Santiago de Compostela, Spain: Springer, 2005, pp. 564–567.
[19] B. Martins and M. J. Silva, "A graph-ranking algorithm for geo-referencing documents," in *Proc. IEEE 13th Int. Conf. Data Mining*, Nov. 2005, pp. 741–744.
[20] F. Melo and B. Martins, "Geocoding textual documents through the usage of hierarchical classifiers," in *Proc. 9th Workshop Geograph. Inf. Retr.*, 2015, Art. no. 7.
[21] F. Melo and B. Martins, "Automated geocoding of textual documents: A survey of current approaches," *Trans. GIS*, vol. 21, no. 1, pp. 3–38, 2017.
[22] S. Roller, M. Speriosu, S. Rallapalli, B. Wing, and J. Baldridge, "Supervised text-based geolocation using language models on an adaptive grid," in *Proc. Joint Conf. Empirical Methods Natural Lang. Process. Comput. Natural Lang. Learn.*, 2012, pp. 1500–1510.
[23] C. Wang, X. Xie, L. Wang, Y. Lu, and W.-Y. Ma, "Detecting geographic locations from Web resources," in *Proc. Workshop Geograph. Inf. Retr.*, 2005, pp. 17–24.

[24] B. P. Wing and J. Baldridge, "Simple supervised document geolocation with geodesic grids," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, 2011, pp. 955–964.
[25] A. G. Woodru and C. Plaunt, "GIPSY: Georeferenced information processing system," *J. Amer. Soc. Inf. Sci.*, vol. 45, no. 9, pp. 645–655, 1994.
[26] Q. Zhang, P. Jin, S. Lin, and L. Yue, "Extracting focused locations for Web pages," in *Proc. Int. Conf. Web-Age Inf. Manage.* Singapore: Springer, 2011, pp. 76–89.
[27] W. Zong, D. Wu, A. Sun, E.-P. Lim, and D. H.-L. Goh, "On assigning place names to geography related Web pages," in *Proc. 5th ACM/IEEE-CS Joint Conf. Digit. Libraries*, 2005, pp. 354–362.

MANSI A. RADKE received the bachelor's degree in computer science from Pune University, India, and the M.S. degree in computer science from the University of Maryland at Baltimore County, Baltimore, MD, USA, in 2008. She is currently pursuing the Ph.D. degree with the Visvesvaraya National Institute of Technology, Nagpur, India. She has been a Faculty Member (Assistant Professor) with the Visvesvaraya National Institute of Technology, Nagpur, since 2012. Her research interests include geographical information retrieval.

NITIN GAUTAM received the master's degree in computer science from the Visvesvaraya National Institute of Technology in 2015. He is currently with Fidelity National Information Services, Pune, India.

AKHIL TAMBI received the master's degree in computer science from the Visvesvaraya National Institute of Technology in 2016. He is currently with Factset Systems India Pvt. Ltd., Hyderabad, India.

UMESH A. DESHPANDE received the B.Tech. degree from the Visvesvaraya National Institute of Technology (VNIT), Nagpur, in 1991, the M.Tech. degree from IIT Bombay in 1993, and the Ph.D. degree in computer science and engineering from IIT Kharagpur in 2005. Since the last 21 years, he has been a Faculty Member (Associate Professor) with the Department of Computer Science and Engineering, VNIT. His areas of interests are distributed systems, networking, artificial intelligence, real-time systems, multi-agent systems, systems biology, and information retrieval.

ZAREEN SYED received the Ph.D. degree in information retrieval from the University of Maryland at Baltimore County, Baltimore, MD, USA. She has published several algorithms for solving real-world problems, such as predicting concepts in documents, modeling user interests, cross document co-reference resolution, linking entities mentioned in text documents to knowledge base entities, interpreting information in tables, unsupervised extraction of structured data, and automatic enrichment of the knowledge base. She is currently with IBM Research as a Watson Research Software Engineer.