An Automatic Approach for Building Place-Name Datasets from the Web

Ying Zhang
North China Electric Power University,
School of Control and Computer Engineering,
Beijing, China
yingzhang@ncepu.edu.cn

Yao-Yi Chiang
University of Southern California
Spatial Sciences Institute
Los Angeles, CA 90089, USA
yaoyic@usc.edu

Craig A. Knoblock
University of Southern California
Department of Computer Science and Information Sciences Institute
Los Angeles, CA 90089, USA
knoblock@isi.edu

Chaopeng Li
Liming Du
Shaowen Liu
North China Electric Power University
School of Control and Computer Engineering
Beijing, China
[chaopeng, duliming, liushaowen]@ncepu.edu.cn

Sanjay Singh
University of Southern California
Department of Computer Science
Los Angeles, CA 90089, USA
sanjaysi@usc.edu

Abstract

A large amount of valuable geographic information about places, including place names, addresses, and telephone numbers exists on the Web but only a small portion of the data is accessible through structured sources (e.g., Google Maps API, GeoNames, and OpenStreetMaps). Moreover, these structured sources only provide static information and do not incorporate changes in a timely manner. This paper presents an approach to automatically build place-name datasets from the results of a Web search engine. In this work, our approach uses the Google search engine API to retrieve webpages associated with specific location names and place types and then parses the returned webpages to extract place names and addresses. The result is a place-name dataset built completely from information on the Web. To evaluate our approach, we collected ground truth data using Google Street View by reading business signs in the image. We tested our approach on 10 city blocks in a U.S. city. The results showed that the proposed approach effectively generated place datasets on a par with Google Maps and outperformed the data available in OpenStreetMap.
Keywords: Geographic information retrieval, place-name dataset, information parsing, visualization

1 Introduction

Place information is important in building a wide variety of geographic applications, such as location-based services for mobile devices. Given these needs, techniques for automatically building place-name datasets are desperately needed. Traditionally, place datasets can be obtained from structured sources such as DBpedia, LinkedGeoData, Wikimapia, Google Places API, and OpenStreetMap. These structured sources provide static information and do not always incorporate the latest changes. Also, commercial sources such as the Google Places API maintain high-quality location data, but many restrictions apply to obtaining and using their data (e.g., usage limits). In contrast, the volume of place information publicly available on the Web is very large and grows rapidly. In addition, the unstructured nature of webpages allows them to change frequently, and up-to-date information about places is often first available on the Web. The challenge is how to extract accurate and timely geographic information from the Web to build place-name datasets.

In this paper, we propose an approach that uses search engines to find relevant webpages for extracting and building place-name datasets. Our approach first searches webpages with a particular location and place types (i.e., <Street Names><City names><Place Types>), such as "Main Street, El Segundo, Coffee Shop". Next, we automatically extract place names and address information from the returned search results. The final outcome is a place-name dataset extracted directly from the Web.

2 Building place name datasets from the Web

Our approach includes four modules, namely, Webpage Collection, Webpage Filtering, Information Extraction, and Visualization. The details of each module are presented in the following subsections.

2.1 Webpage Collection

In this work, we use businesses as the target place types to extract from the Web. To collect relevant business webpages, we first compile a list of search keywords for querying a search engine API. The search engine queries include three parts, namely, Street Name, City Name, and Business Type. To prepare our queries, we first set a specific city name, and

the Webpage Collection module automatically downloads street data from OpenStreetMap (OSM) and extracts street names from the OSM "addr:street" attribute (e.g., "Main Street" and "N Sepulveda Boulevard"). For business types, we manually prepare a list of popular place types, such as Restaurant, Bar, Hotel, Store, Theater, Hair Salon, etc. Finally, the Webpage Collection module sends the compiled queries to a search engine for finding business webpages. We use the Google search engine for this task because of its high-quality search results.

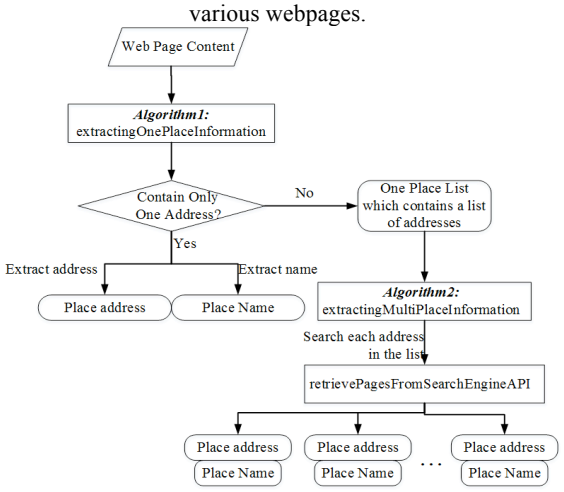
2.2 Filtering Irrelevant Pages

Search engines can return as many results as we want, but most of them are irrelevant. In the Webpage Collection module, the collected webpages include many real estate listings (which contain mostly residential addresses). To filter out these listings, in this module, we first query the Google search engine using a recently sold home address (which can be obtained from any real-estate website) and then use the returned results to find the domain names of popular real-estate websites. For example, a Google search with "2117 Tondolea Ln" returns webpages from real estate websites including Zillow, Redfin, Movoto, etc. Using these websites, we learn the URL patterns of real-estate websites such as "www.zillow.com" and "www.redfin.com". Next, we use these learned URL patterns to remove real-estate websites automatically.

2.3 Extracting Place Information

Because webpages have various document structures. Extracting address information from these different types of webpages is challenging. For instance, some of the collected webpages each represent a specific business, from which a single address can be extracted; whereas other results are yellow-page like listings describing multiple venues. Figure 1 shows the overall process of our approach for place information extraction. If Algorithm 1 determines that a webpage is for a particular business, it extracts a single place name and a unique address. Otherwise, the webpage is passed to Algorithm 2 for extracting multiple place names and addresses.

Figure 1: Extracting geospatial information of places from various webpages.



In Algorithm 1, there are two cases in the address extraction: the first case is based on the condition that the entire address exists on one line (in the webpage); the second one applies to the situation where the address spans across multiple lines.

For the first case (lines 3-6), we first find the line in the webpage that starts with a number (line 3) and contains the city name (line 4). In addition, the length of the line should be less than a given threshold (line 5) assuming that the probability of a line containing an address is relatively low if the length of the line is long. In this paper, the threshold value

Algorithm 1: extractingOnePlaceInformation

```
Input  :Web Page page, City Name cityName,
        Maximum length of Address threshold
Output : singlePlace<address, name>,
        List multiAddressList
1  content←FilterHtmlTag(page)
2  while line1 in content do
3    if StartWithNumber(line1) then
4      if StrContainCityName(line1, cityName) then
5        if theLengthOf(line1)<threshold then
6          multiAddressList.append(line1)
7      tempValue←line1
8    else if StartWithCityName(line1, cityName) then
9      line2← catenation(tempValue,line1)
10     if StartWithNumber(line2) then
11       if StrContainCityName(line2, cityName) then
12         if theLengthOf(line2)<threshold then
13           multiAddressList.append(line2)
14   if addr_list.hasOnlyOneAddress() == true then
15       singlePlace.address← multiAddressList.firstElement
16       singlePlace.name ← extractingWebPageTitle(page)
17       return singlePlace
18   else
19     return multiAddressList
```

Algorithm 2: extractingMultiPlaceInformation

```
Input  : List addr_list
        //A list of addresses from one page
Output : PlaceList<address, name>
1  while item in addr_list do
2    pList←retrievePagesFromSearchEngineAPI(item)
3    //Retrieve the returned pages from search Engine.
4    while page in pList
5      if page.containOnlyOneAddress==true then
6        addr←extractOneAddress(page)
7        if addr==item then
8          PlaceList.address← addr
9          PlaceList.name←extractingWebPageTitle(page)
10         break
11 return PlaceList
```

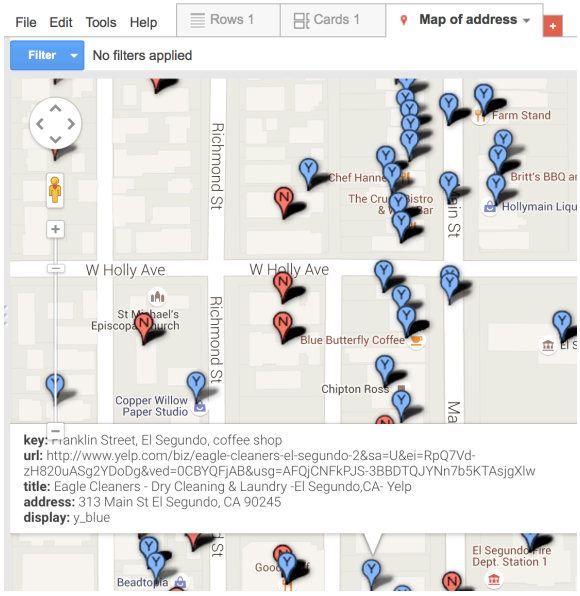
is set to 100 empirically. For the second case (lines 8-13), we use a similar method to determine whether or not a combination of two lines represent one address. For example, if the first line starts with a number and the second line contains city name, both of them are merged and extracted as an address if the length of the combined line is shorter than the threshold. In Algorithm 1 when the webpage contains only one address, we assume the webpage title is the place name. If the page contains multiple addresses, Algorithm 1 sends a list of addresses to Algorithm 2.

In Algorithm 2, we search each address in the address list using the search engine API (line 2). Among all the returned webpages, if a returned webpage contains only one address and the address is the same as the address used to query the search engine API, the corresponding webpage title is considered as the place name. The webpage title together with the extracted address is used to describe the place (line 5-10).

2.4 Visualization

Once we extract the place names and place addresses in the third module, we use a geocoding tool (e.g., Google Fusion Tables[1]) to convert the place addresses into geographic coordinates for visualizing the extracted place dataset. Google Fusion Tables is an experimental data visualization web application to gather, visualize, and share data tables. As illustrated in Figure 2, we customized two kinds of pins in our work. The blue pins with label 'Y' represent the place names with business information obtained from the processes in Sections 2.1-2.4. The red pins with label 'N' denote those places for sale or rent, which are filtered out by the Webpages Filtering module.

Figure 2: Geocoding and visualizing the extracted place dataset.



3 EXPERIMENTAL RESULTS

This section describes our experimental setup and presents the results of our evaluation. We built a place-name dataset form the Web for the city of El Segundo (California) using the approach described in this paper and compared our datasets with OpenStreetMap and Google Maps. We evaluated the three datasets in terms of precision, recall, and F-score [1]. We manually collected ground truth data by looking at the Google Street View images to find business locations and names.

3.1 Experimental Settings

Table 1: The search query settings in El Segundo

Street Names (in El Segundo) ***(Extracted from OpenStreetMap)***	Richmond Street
	Grand Avenue
	Main Street
	Franklin Street
	South Sepulveda Boulevard
	Rosecrans Ave
	N Sepulveda Boulevard
	Indiana Street
	Penn Street
	Center Street
	Sheldon Street
	Illinois Street
City Names	El Segundo
Place Types	Restaurant/ Bar/ Gym/ Store/ Theater/ Hair Salon/ Coffee Shop/ Tutoring/ Realtor/ Hotel/ Church/ Hospital/ Bank/ Club/ School

In our experiments, we first queried the Google search engine to find relevant business webpages for the city of El Segundo. Table 1 gives the search queries used in our experiments. Street names were obtained from the OpenStreetMap automatically (Section 2.1). In this experiment, we selected ten blocks in El Segundo to evaluate the performance of the presented approach.

3.2 Evaluation Measures

To evaluate our approach, we manually looked at the Google Street View to collect the ground truth data by reading signs visible in the imagery. It could be the case that Google Street View itself was not up-to-date, but this temporal bias should affect all three test datasets (datasets from our approach, from Google Maps, and from OpenStreetMap). Three measures, precision, recall, and the F-measure were used to evaluate the performances of the presented approach in this paper.

3.3 Experimental Results

Table 2 shows an example of the extracted places from the Web using our approach, places from Google Maps and OpenStreetMap, and the ground truth in one of the ten tested blocks. In this block, OpenStreetMap returned only one place name. Our extracted dataset contained one more place than Google Maps. The row in yellow shows a correctly extracted place using our approach, which was missing in the Google Maps dataset.

[1] https://support.google.com/fusiontables/answer/2571232

Table 2: Example test results and the ground truth for one city block.

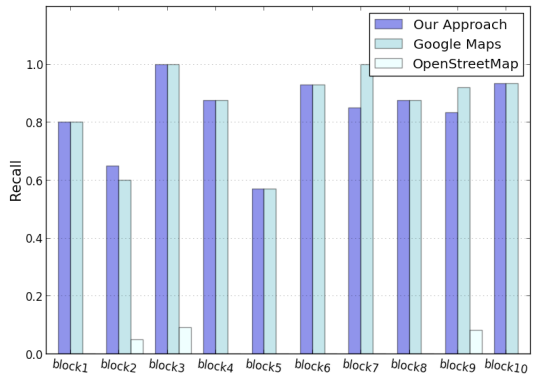
Our approach		Google Maps		Google Street View (Ground Truth)		OpenStreetMap	
Place Name	Address	Place Name	Address	Place Name	Address	Place Name	Address
Main StreetRealty - Real Estate Agents	361 Main St El Segundo, CA	Main StreetRealty	361 Main St El Segundo, CA	Main Street Realty	361 Main St El Segundo, CA	---	---
On The Main-El Segundo	353 Main St El Segundo, CA	On the main	353 Main St El Segundo, CA	On the main	353 Main St El Segundo, CA	---	---
Blue Butterfly Coffee Co - Coffee & Tea	351 Main St, El Segundo, CA	Blue Butterfly Coffee	351 Main St, El Segundo, CA	Blue Butterfly Coffee Company	351 Main St, El Segundo, CA	---	---
Chipton Ross in El Segundo | Chipton Ross	343 Main St El Segundo, CA	Chipton Ross	343 Main St El Segundo, CA	Chipton-Ross	343 Main St El Segundo, CA	---	---
The Jewelry Source - Jewelry	337 Main St El Segundo, CA	Jewelry Source	337 Main St El Segundo, CA	The Jewelry Source	337 Main St El Segundo, CA	---	---
The Original Rinaldi's Sandwiches	323 Main St El Segundo, CA	The Original Rinaldi	323 Main St El Segundo, CA	The Original Rinaldis Italian Dcli	323 Main St El Segundo, CA	---	---
Steve's Burgers - Burgers	321 Main St El Segundo, CA	Steve's Burgers Plus	321 Main St El Segundo, CA	Steve's Burgers Breakfast¡¤Lunch¡¤ Dinner	321 Main St El Segundo, CA	---	---
Eagle Cleaners - Dry Cleaning & Laundry	313 Main St El Segundo, CA	Eagle Cleaners	313 Main St El Segundo, CA	Cleaners eagle	313 Main St El Segundo, CA	---	---
World Karate - Martial Arts	309 Main St El Segundo, CA	---	---	World Karate	309 Main St El Segundo, CA	---	---
Bank of America (ATM)	343 Main St El Segundo, CA	Bank of America (ATM)	343 Main St El Segundo, CA	Bank of America ATM	343 Main St El Segundo, CA	Bank of America	343 Main St El Segundo, CA

Note: The row in yellow shows a place extracted from the Web and found in the ground truth, but was missing from other sources.

Table 3: The overall results for all the test blocks

Block No.	Our Approach		Google Street View (Ground Truth)	Google Maps		OpenStreetMap	
	Total number of the extracted records	Number of the exact records	Total number in Ground Truth	Total number of the extracted records	Number of the exact records	Total number of the extracted records	Number of the exact records
1	13	12	15	12	12	0	0
2	13	13	20	12	12	1	1
3	11	11	11	11	11	1	1
4	7	7	8	7	7	0	0
5	5	4	7	4	4	0	0
6	11	11	13	11	11	0	0
7	14	13	14	14	14	0	0
8	7	7	8	7	7	1	0
9	11	10	12	11	11	1	1
10	14	14	15	14	14	0	0

Figure 3: The recall comparison between our approach, OpenStreetMap, and Google Maps



Note: The recall of our approach was much higher than that of OpenStreetMap. The proposed method performed equal to or slightly lower than Google Maps for most of the test blocks except for block 2 (where our method outperformed Google Maps).

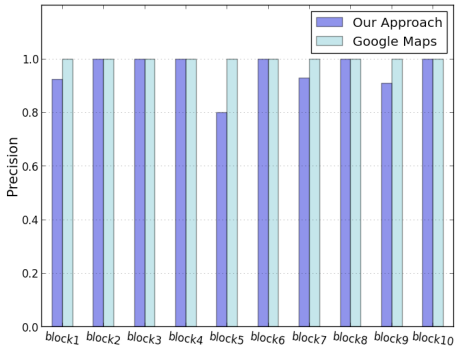
Table 3 presents the overall results for all test blocks. We compared our approach with Google Maps and OpenStreetMap regarding the recall, precision, and F-Score on the ten test blocks (Figures 3 - 6) . Figure 3 shows that the recall of our approach was much higher than OpenStreetMap and was equal to or slightly lower than Google Maps for most of the test blocks except for block 2 (where our method outperformed Google Maps).

The average precision numbers for Google Maps, OSM, and our approach were 100%, 75%, and 95.6%, respectively. Figure 4 illustrates that the precision of our approach was as high as that of Google Maps on 6 out of the 10 test blocks. Since OSM returned zero places in most of the test blocks, we did not show its precision here. For the areas such as El Segundo where OSM does not have much business information, our results could be used to provide a starting point for crowdsourcing. Comparing with our approach, Figure 5 shows the number of the missing places on OpenStreetMap in each block.

Figure 6 shows our approach outperformed OpenStreetMap in the F-score and obtained a similar F-score as Google Maps. The precision of our approach was not as high as that of Google Maps when the number of the extracted places was large (more false-positives). The false-positives in our approach could be for two reasons. First, we used the webpage titles as the place names, but a few of the extracted

places used addresses as their webpage titles. About 60% of the incorrect results were caused by the non-place-name webpage titles. Second, we extracted addresses from short strings that start with numbers and contain the city name. Most of the addresses satisfied this rule, but there were some exceptions. For example, our approach incorrectly extracted the place name "99 Cent Store - Discount Store - El Segundo, CA" as an address. About 40% of the incorrect results was caused by having incorrect addresses. We would address the first problem by filtering out those webpage titles that are in the form of an address. To overcome the second problem, we would add additional conditions such as whether or not the address line contains a street name to determine the correct place addresses.

Figure 4: The precision comparison between our approach and Google Maps.



Note: Our approach achieved the same precision as Google Maps on 6 out of the 10 blocks.

Figure 5: Number of place names missing from OpenStreetMap (compared to our approach and ground truth) in each of the test blocks.

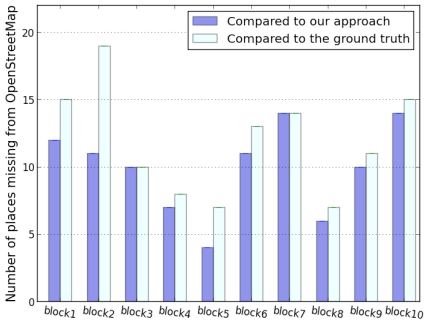
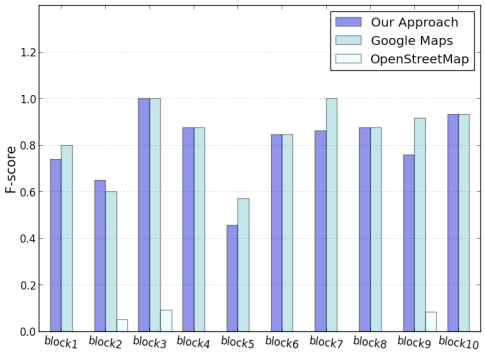


Figure 6: The F-score comparison between our approach, OpenStreetMap, and Google Maps.



Note: Our approach outperformed OpenStreetMap completely in F-score and could obtain high F-score as Google Maps did on 6 out of the 10 test blocks. On block 2, our approach outperformed Google Maps.

Compared with the ground truth, 90% of the missing places in our approach were due to the lack of search query varieties (i.e., the searched business types do not cover every business type in the test blocks). For example, in our experiment, since we did not use "music venue" as a place type in the search query, "Old Town Music Hall", "South Bay Customs", and etc., were missed in the search results. The rest of the missing places was caused by missing webpages (i.e., the business did not have a webpage). We could mine place types from webpages to add new varieties of search queries automatically for improving our recall in the future

4 RELATED WORK

As the volume of spatial information on the Web grows daily, exploiting publicly available Internet sources for extracting place information can help to quickly and effectively generate a large set of place information for any city on Earth.

A number of researchers have collected data for generating gazetteers from structured Internet sources. The approach presented in [2] extracted points of interest from a set of popular Web sources including DBpedia, OpenStreetMap, Wikimapia, Google Places, Foursquare, and Eventful. The first two sources provide SPARQL endpoints, and the latter four sources offer RESTful API. In addition, Gelernter et al. [3] discussed a method to enrich a gazetteer by identifying sources of novel local gazetteer entries in crowdsourced OpenStreetMap and Wikimapia geotags.

Considering the dynamic nature of place datasets, one could resort to Web scraping, i.e., parsing the HTML code of the webpages for generating place datasets that reflect the latest place information on the Web. A considerable number of researchers have used search engines to query the Web as a source to extract information for generating place names [4, 5]. Blessing and Schütze [6] use the Google API to query the Web and generate place-name variations. Their queries includes the names of places as the positive terms and some stopwords as negative terms. The stopwords are used to exclude webpages that make the query result noisy. Brindley et al. [7] discover neighborhood place names from addresses found in web pages. They extract postal addresses from the Web and create relatively simple linguistic models to produce neighborhood definitions that are both probabilistic and dynamic. Popescu et al. [8] present an automated technique for creating and enriching a geographical gazetteer. Their technique merges disparate information from Wikipedia, Panoramio, and Web search engines to identify geographical names. Uryupina [9] presents an approach to the automatic

acquisition of geographical gazetteers from the Internet. A new gazetteer can be learned from a small set of pre-classified examples by applying bootstrapping techniques.

Most approaches related to place-name-based information retrieval from the Web need to deal with the ambiguous place names [10]. The first type of ambiguity is called multiple references, in which the same name can refer to various places. The second type is called a variant name, which corresponds to when a place is given different names. The third one is the geo/non-geo ambiguity [11], which represents place names with common, not place words. Several methods are presented to deal with ambiguity and improve the performance of extracting place names. Santos et al. DeLozier et al. [12] present a toponym resolver that uses the profiles of the local clusters to build a system that grounds toponyms by finding areas of overlaps in the distributions of toponyms and other words in a toponym's context. In contrast, we present a simpler method in this work to avoid ambiguity problem by using queries of <Street Names><City names><Place Types>. Because the three components in our queries mutually restrict each other (given the specific address and the place type, the possibility that the extracted place names are ambiguous is small), our approach does not suffer from the problem of ambiguous place names.

5 DISCUSSION AND FUTURE WORK

In this paper, we presented our approach for building place-name datasets from the Web. Our evaluation showed that although the precision of our approach could be lower than Google Maps when the number of extracted places was large, the recall of our approach was consistently similar to or higher than Google Maps.

Our current and future work in this area focuses on three main directions. The first is to improve the accuracy of the Information Extraction module. This can be done by exploiting the DOM structure of a webpage. The second direction is to improve the recall by automatically mining keywords of place types from the Web. We can also take advantage of the ontology concepts to define a flexible way to establish semantically rich relationships between place types. The rich relationships between place types can provide better search queries to search engine and thus improve the recall. Lastly, we plan to conduct a more extensive experiment that covers cities of various types in the world.

6 ACKNOWLEDGMENT

This work was jointly supported by the National Natural Science Foundation of China (No.61305056, No.61300132, No. 61403137), Beijing Higher Education Young Elite Teacher Project (No.YETP0702, No.YETP0706), Overseas Expertise Introduction Program for Disciplines Innovation in Universities(Project 111) (No. B13009), and the Fundamental Research Funds for the Central Universities (No.2015MS35, No. 2015MS28).

7 REFERENCES

[1] Muslea, I., Minton, S.N., Knoblock, C.A.. Active learning with strong and weak views: a case study on wrapper induction, In IJCAI'03: Proceedings of the 18th international joint conference on Artificial intelligence, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp.415–420, 2003.

[2] Lamprianidis, G., Skoutas, D., Papatheodorou, G., Pfoser, D.. Extraction, integration and analysis of crowdsourced points of interest from multiple web sources. In: Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Crowdsourced and Volunteered Geographic Information, pages 16–23, 2014.

[3] Gelernter, J., Ganesh, G., Krishnakumar, H., Zhang, W.. Automatic gazetteer enrichment with user-geocoded data. In Proceedings of GEOCROWD '13, New York, NY, USA, pp.87–94, 2013.

[4] Keßler, C., Janowicz, K., Bishr, M.. An agenda for the next generation gazetteer: geographic information contribution and retrieval, Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, November , 2009, Seattle, Washington, pp.4-6.

[5] Purves, R., Clough, P., Joho, H.. Identifying imprecise regions for geographic information retrieval using the web. In: GIS Research UK 13th Annual Conference, pp.313–318,2005.

[6] Blessing, A., Schütze, H.. Automatic acquisition of ver-nacular places. In Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services (iiWAS '08), pp.662–665, 2008.

[7] Brindley, P., Goulding, J., Wilson, M.L.. A data driven approach to mapping urban neighbourhoods, Proceedings of the 22th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, November pp.04-07, 2014, Dallas, TX, USA.

[8] Popescu, A., Grefenstette, G., Moëllic, P.-A.. Gazetiki: automatic creation of a geographical gazetteer, Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries, June, Pittsburgh PA, USA, pp.16-20, 2008.

[9] Uryupina, O., Semi-supervised learning of geographical gazetteers from the internet, In Proceedings of the HLT-NAACL 2003 Workshop on Analysis of Geographic References, pp.18–25, 2003.

[10] Volz, R., Kleb, J., Mueller, W.. Towards ontology-based disambiguation of geographical identifiers. In: Proc. of the 16th WWW Conference, Banff, Canada, 2007.

[11] Amitay, E., Har'El, N., Sivan, R., Soffer, A.. Web-a-where: geo- tagging web content. In: Proceedings of the 27th annual interna- tional ACM SIGIR conference on research and development in information retrieval, SIGIR'04. ACM, New York, pp.273–280, 2004.

[12] Delozier, G., Baldridge, J., London, L.. Gazetteer-Independent Toponym Resolution Using Geographic Word Profiles, In Proceedings of AAAI 2015, The AAAI Press.