Pattern-Based Extraction of Addresses from Web Page Content

Saeid Asadi, Guowei Yang, Xiaofang Zhou, Yuan Shi,
Boxuan Zhai, and Wendy Wen-Rong Jiang

School of Information Technology & Electrical Engineering
The University of Queensland
GP78 South, St. Lucia, Brisbane, Australia
{asadi, guowei, zxf, yuanshi, bxzhai, wjiang}@itee.uq.edu.au

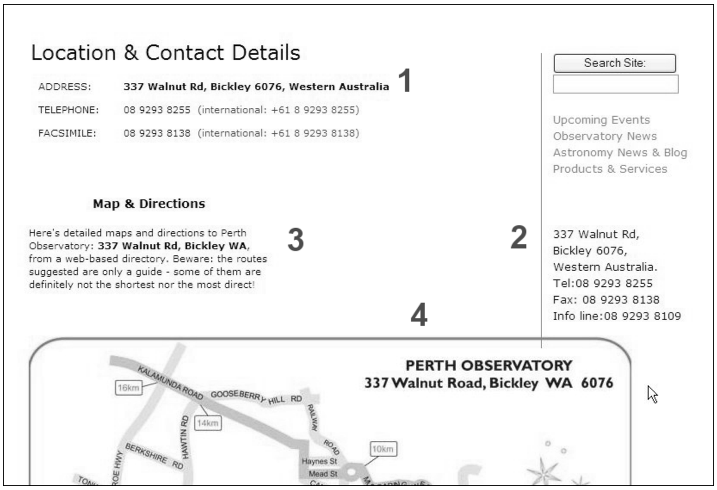
Abstract. Extraction of addresses and location names from Web pages is a challenging task for search engines. Traditional information extraction and natural processing models remain unsuccessful in the context of the Web because of the uncontrolled heterogenous nature of the Web resources as well as the effects of HTML and other markup tags. We describe a new pattern-based approach for extraction of addresses from Web pages. Both HTML and vision-based segmentations are used to increase the quality of address extraction. The proposed system uses several address patterns and a small table of geographic knowledge to hit addresses and then itemize them into smaller components. The experiments show that this model can extract and itemize different addresses effectively without large gazetteers or human supervision.

Keywords: Address Extraction, Web page Analysis, Address Itemization.

1 Introduction

Search engines are the main tools for searching the resources on the World Wide Web. Although general search engines are widely used to retrieve Web pages, they fail to handle many queries according to the users' needs and interests. Specialized search engines instead are dedicated to find either particular types of resources such as sounds, photos and movies or Web pages based on different criteria e.g. language or geographic location.

Geographic or location-based search has recently received attention from both academic and industrial groups. The popularity of the Web originally was because of providing access to the resources regardless of their physical locations. However, parallel to this global accessibility, many services and facilities have become available on the Web which refer much more to specific locations. People use search engines to find Web pages of local services and events around them or in a particular area. They search for restaurants, accommodation, governmental offices, clubs and social activities on the Web. While a significant proportion of the Web queries contain geospatial dimensions and refer to specific locations,


Location & Contact Details

ADDRESS: **337 Walnut Rd, Bickley 6076, Western Australia** 1

TELEPHONE: 08 9293 8255 (international: +61 8 9293 8255)

FACSIMILE: 08 9293 8138 (international: +61 8 9293 8138)

Map & Directions

Here's detailed maps and directions to Perth Observatory: **337 Walnut Rd, Bickley WA**, from a web-based directory. Beware: the routes suggested are only a guide - some of them are definitely not the shortest nor the most direct! 3

2

337 Walnut Rd,
Bickley 6076,
Western Australia.
Tel:08 9293 8255
Fax: 08 9293 8138
Info line:08 9293 8109

4

PERTH OBSERVATORY
337 Walnut Road, Bickley WA 6076

Fig. 1. Different ways of mentioning an address in a Web page

general search engines can not differentiate these queries and handle them properly [15,2]. Few commercial geographic search engines have been commercially developed among them Google Map [1] and Yahoo Local [2] are notable which are geographically limited and they often search commercial databases such as Yellow Pages instead of covering real Web pages. The problem basically originates from the ambiguous dynamic nature of location names, various addressing styles, lack of geographic information, and multiple locations related to a Web resource.

Geographic Web search has its own specifications which makes it different from general search. A brief of the location-based search engine architecture and tasks can be found in [7,11,1]. Basically, a geographic search engine must be able to find related addresses and location names and assign them to Web pages. Current address extraction techniques basically require large gazetteers which are expensive and unavailable for many countries.Also different markup styles e.g. HTML, XML and DOM increase the complexity of address extraction on the Web. As a result, natural language processing models are not able to extract all addresses and location names from Web page contents. Figure 1 shows different styles of mentioning an address in a typical Web page. The equivalent HTML code of these addresses are different and as a result the extracted addresses are not identical.

In this paper, we introduce a pattern-based model which uses HTML and visual segmentations to improve address extraction on Web pages. Our model detects an address and itemizes it to its components using patterns and a small set of triggers and location names. Our contribution to address extraction is trying to divide an address to its semantic components and also eliminate large scale gazetteers. This model is automatic and does not require expensive gazetteers

[1] http://maps.google.com/

[2] http://local.yahoo.com/

or much human effort. It also accumulates new location names to its knowledge repository.

In section 2, we review the previous work on information extraction and geo-tagging of Web pages.In sections 3 and 4, the general architecture and the details of the proposed model are described in depth respectively. The experimental setup and results are discussed in sections 5.

2 Background

Information Extraction (IE) systems extract the information about pre-specified types of events, entities or relationships from a text and this information is often added to a database for more applications. IE requires knowledge about proper names. In address extraction, this knowledge comes normally from gazetteers. Several works have focused on the use of supervised learning techniques such as Hidden Markov Models [10,17] and rule learning [18]. These techniques learn a language model or a set of rules from a set of hand-tagged trained documents; then apply the model or rules to new texts. Models learned in this manner are effective on documents similar to the set of training documents, but they are ineffective when applied to documents with a different genre or style. As a result, this approach has problem scaling to the Web due to the diversity of text styles and genres on the Web and the prohibitive cost of creating an equally diverse set of manually tagged documents. The experiments on extraction of location names with language patterns and without gazetteers such as [12] often end in bad results because of the ambiguities in geographic names.

The geography or location of Web resources has been measured as *geographical scope* or the location of Web pages that link to a page [8], *target location* or the location of people who visit a Web page [3], and *geographic focus* or a location which most of the content or addresses on a page refer to [1]. A review of different sources of geographic information in Web pages can be found on [11]. Geographic features e.g postcodes and telephone codes haven been used e.g. in [16,4] for geo-tagging. In [6], the visual structure of addresses, as they are shown to users, has been used to detect and extract address blocks from Web pages.

Gazetteer approach [14] and pattern-based models [2] often lead to weak results for extraction of geographic features on the Web. Web-a-Where [1] uses gazetteers to find and disambiguate location names on Web pages. Their results indicates that having all location names in the database does not insure a full coverage. KnowItAll [9] construct extraction rules based on the instants found in documents. Then the system decides whether a name belongs to a particular class or not using statistical rules. Similarly, Ourioupina [13] reports an algorithm for geographic knowledge acquisition in which a location name is added to different patterns and sent to search engines. Based on the frequency of the results, the best class is selected for the candidate. For example, the phrase *"city of Paris"* seems to be more repeated on the Web than *"cities in Paris"*. As a result, the system decides that Paris is a CITY not a COUNTRY.

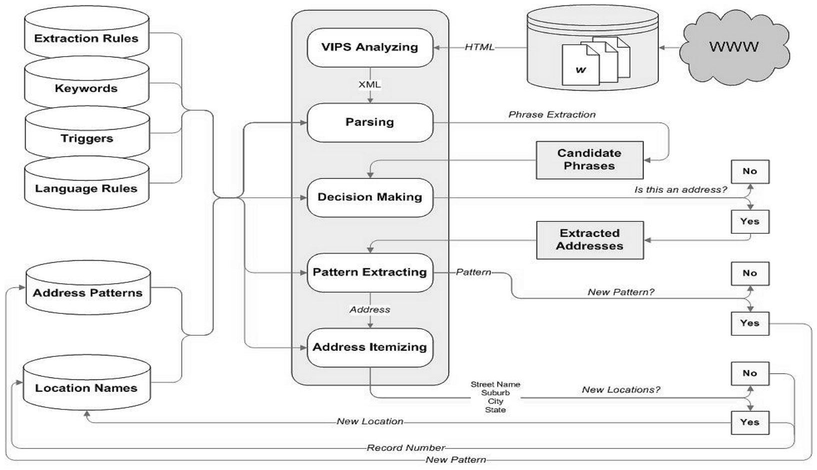


Fig. 2. An overview of the address extraction system described in this paper

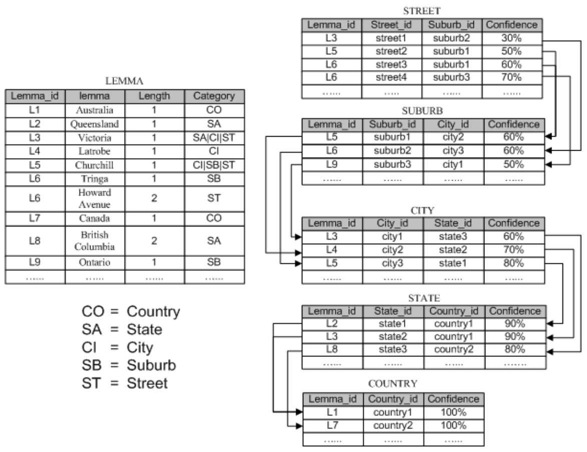
3 The System Architecture

The proposed address extraction system consists of five components: HTML Pre-Processor, Parser, Knowledge Searcher, Decision Maker, and Knowledge Accumulator. A general picture of the proposed system is shown in Figure 2.

3.1 HTML Pre-processor

The content of Web pages are embedded in HTML tags. An effective information extraction model should analyze HTML properly. By parsing HTML alone, many pieces of geographic information which are embedded in separate lines (Fig. 1) or tabular structures can be lost. An information extractor parses a file line by line and in this case, it can not find the component of each address properly. To overcome this issue, we convert HTML files to XML in a way that the visual structure of Web pages will be considered. The system first transforms a HTML file into a XML file by employing the VIPS Demo software developed by Microsoft Research Group [3]. Vision-Based Page Segmentation (VIPS) [5,20] is a technique for automatic segmentation of Web page into smaller blocks according to the visual segmentations that a user can see on a Web page. The next step is in-depth analyzing and traversing the XML to obtain content information from every XML leaf node, and sorting them in a linear sequence together with their node numbers. At the same time, a node index is built where children nodes and their parents are connected together with bidirectional pointers which later can support trace of the parent node and then sister nodes to get a whole view of a complete address.

[3] For more details see: http://www.ews.uiuc.edu/ dengcai2/VIPS/VIPS.html



LEMMA

Lemma_id	lemma	Length	Category
L1	Australia	1	CO
L2	Queensland	1	SA
L3	Victoria	1	SA|CI|ST
L4	Latrobe	1	CI
L5	Churchill	1	CI|SB|ST
L6	Tringa	1	SB
L6	Howard Avenue	2	ST
L7	Canada	1	CO
L8	British Columbia	2	SA
L9	Ontario	1	SB
......	......	......	......

CO = Country
SA = State
CI = City
SB = Suburb
ST = Street

STREET

Lemma_id	Street_id	Suburb_id	Confidence
L3	street1	suburb2	30%
L5	street2	suburb1	50%
L6	street3	suburb1	60%
L6	street4	suburb3	70%
......	.....	......	......

SUBURB

Lemma_id	Suburb_id	City_id	Confidence
L5	suburb1	city2	60%
L6	suburb2	city3	60%
L9	suburb3	city1	50%
......	......	......	......

CITY

Lemma_id	City_id	State_id	Confidence
L3	city1	state3	60%
L4	city2	state2	70%
L5	city3	state1	80%
......	......	......	......

STATE

Lemma_id	State_id	Country_id	Confidence
L2	state1	country1	90%
L3	state2	country1	90%
L8	state3	country2	80%
......	......	......	....

COUNTRY

Lemma_id	Country_id	Confidence
L1	country1	100%
L7	country2	100%
......	......	......

Fig. 3. A view of the Knowledge-Base (KB)

3.2 XML Parser

The XML Parser is responsible for reducing the search scope from a whole file to the areas of interest by defining syntactic filtration. It tries to find all candidate phrases (potential addresses) in a node. Besides, the parser also divides a potential address into its component inferred from content clues such as *comma, <br>* etc. Each word of an address element is coarsely classified into different categories (We have defined these categories: *NUM, Trigger, NOTICE, Cased Word, Preposition, Period Ended, quotes Ended*). Each segment obtained in this step, will be utilized as default searching unit of Database Searcher.

3.3 Knowledge Searcher

Knowledge Searcher itemizes elements of a potential address according to the accumulated knowledge and patterns Knowledge-Base (KB) (Figure 3). It finds all possibilities of a potential address and forms them into a list of possible patterns in three steps:

Standardizing Word Formats. Many words and names have different spells (e.g. *Tehran* and *Teheran*), abbreviations (e.g. *street* and *st.*) and name variations or synonyms (e.g. *Netherlands* and *Holland*). Knowledge Searcher first standardizes or converts different variations to their pre-selected formats. For example, wherever there is *QLD* in an address, the system considers it as Queensland. This is not always correct since variations or abbreviations of some words may also be part of other names. For example, *St* is a common abbreviation for street in addresses. However, it is also a common abbreviation for Saint in many people or place names. The Knowledge Searcher retains the original spelling as well as the standardized format, and both will be looked up in the Knowledge-Base step later.

Knowledge-Base Place Name Matching. Although the parser divides addresses into their elements, confusing formats of writing addresses in HTML files still remain. Many place names are written in sequence without any separation symbol. The Knowledge Searcher separates elements into more delicate level based on the pre-accumulated knowledge in Knowledge-Base (KB).

Ambiguity Eliminating. Knowledge Searcher tries to match place name as long as possible. For example, if *St Lucia* has already been stored in database as a unique name, then any immediate co-occurrence of *St* and then *Lucia* will be considered as *"St Lucia"* instead of *St <Trigger>+ Lucia <Unknown word>*.

3.4 Decision Maker

Decision Maker determines whether a candidate phrase is an address or not by matching it with address patterns already stored in a database. A matched pattern with the highest confidence score will be chosen for each candidate. If the the phrase exceeds a threshold, it will be confirmed as an address, and then it will be itemized to its components. The decision maker basically performs three tasks: 1) Delimitating ambiguities and conflicts of place names; 2) Itemizing each potential address to its elements; and 3) Adding the lost parts to address based on a location tree wherever it is possible. For example, the address *"No. 10, William Street, Toowong, Queensland"* will be modified as *"No. 10, William Street, Toowong, Brisbane, Queensland, Australia"* in it Brisbane and Australia has been added to the address according to a location tree.

3.5 Knowledge Accumulator

Knowledge Accumulator is the last component of the system. Knowledge accumulation exhibits in two aspects:

Location Accumulation. Place names are collected based on the addresses recognized in previous part. Knowledge Accumulator estimates whether an unknown word or phrase (few words) can be considered as a place name or not, and if yes, which type of place it should be classified to, and what confidence score should be assigned to it. Besides, it recomputes (increase/decrease) confidence scores of known place names.

Address Pattern Accumulation. The system gathers new patterns and increases their confidence score with next occurrences of them. This step is similar to [19] which automatically builds a gazetteer by using new locations extracted through patterns.

4 Details of Our Work

4.1 Ambiguity Elimination

There are often two types of ambiguities in addresses: syntactic and semantic. *Syntactic ambiguity* comes from the fact that one word may have different spellings or abbreviations e.g. *"St"* for *street*and *Saint*. The lemma *"St"* has at least two explanations. The second ambiguity, *semantic*, resides in different understandings of a known string. Take lemma *"Sydney"* for example. The famous Sydney is the capital city of Australia; but, it also refers to few other places in USA, Canada etc. Such geo/geo ambiguities happen more pervasively in suburb and street levels. Besides, geo/non-geo ambiguities are also common: *Sydney* is also a common female name in several countries.

Syntactic ambiguity can be avoided by combining a word with its previous and following words and search them in Knowledge-Base (KB). The method is carried out following the heuristic that the longer a word list can be matched in KB, the more chance to be a unique name. For example, having in phrase "... Shopping Center St Lucia..." if *St Lucia* is found in KB, then "St" will be interpreted as "Saint" instead of *"street"*. Semantic ambiguity is solved based on Pattern Base(PB) and Knowledge-Base(KB). All possible permutations of different identities of a known place names in a potential address can be obtained by searching KB. A confidence score is calculated for each permutation by matching them with patterns. The permutation with the highest confidence will be chosen.

The next step is finding all possible paths that connect identities of every two adjacent place names in a potential address. If a place name in a path cannot be connected to its neighbors with any of its known identities, it will be labelled as *Unknown Word*. Pattern-based heuristic is then applied to calculate confidence of each possible path. If there is a matched pattern, its confidence will be used to make a final decision by considering both pattern confidence and identity confidence.

Example: Assume that there are 9 lemmas in KB that cover all the levels of the address in Figure 3. Among them, 3 lemmas have multiple identities. *"Victoria"* is a street name in different cities, a state of Australia and also a city in Canada. *"Churchill"* can be a city, a suburb and also a street. *"Howard Avenue"* refers to few streets of different cities in Australia and other countries.

Following algorithm indicates how place names are detected in Phrases. Let W_i to be the i*th* word in a candidate phrase:

```
Place Name Detecting Algorithm.
input: PW - A candidate phrase
        W_i - the ith word in PW
        f - any syntactic format of W_i
        KB - Knowledge-Base
        C_i - Result Collection
1. PW(pre_word, W_i) {
2.   if ((pre_word + f) = a place name found in KB)
3.      add (pre_word + f) to C_i;
4.   if (pre_word + f) = part of a name in KB
5.      pre_word = pre_word + f;
6.      PW(pre_word, W_i+1);//try next word in PW
7. }
```

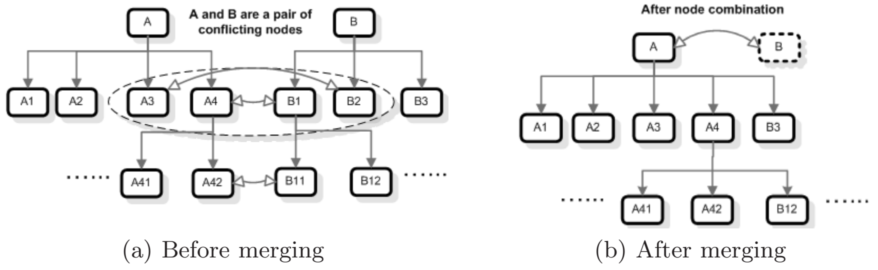


Fig. 4. Merging Conflict Nodes: (a)Before merging (b)After merging

Based on PW(), the syntactic ambiguity elimination function is like the following:

Syntactic Ambiguity Elimination.

```
1. SyntacticAE(Potential) {
2.   current_word = first word in Potential
3.   C = NULL; //initialize C
4.   While current_word != EOF
5.   {
6.     C = SAE (C, current_word); //add longest result in C
7.     current_word = next new word in Potential;
8.   }
9. }
```

4.2 Conflict Elimination

Address Conflict refers to inconsistencies between accumulated knowledge in KB and extracted information from the Web. According to our experience, there are two causes of address conflict. One comes from word misspelling and synonymy. *"Sydny"* for example, is a misspelling of *Sydney.* The other conflict comes forth due to incompleteness of KB e.g. *"Newcastle"* is recorded as a city in Australia in KB while it is also a city in UK not covered in KB. Conflict elimination is performed via a self-learning process. Since it is error-prone for us to determine at a early stage which type of conflict it belongs to, and what to do with it, we delay making decision to later. Figure 4 illustrates how the system deals with conflicting nodes.

1. Keeping the Conflict: Conflicting pairs of *conflicting element* and *original element* in a potential address are recorded in a *conflicting table.* Then, a new branch of address for the conflicting element will be built in KB *conflicting element.* All the place names in the result address that belongs to original element will be issued with new identities and stored as new nodes in the new branch with relatively low confidence. If this branch is meaningful, it will appear next time and its confidence will increase accordingly. Otherwise, it will have the same low score.

2. Removing Meaningless Conflict Element: Regular work of conflict checking is carried out after a certain period. First, the confidence of each conflicting node is checked. If the confidence is still low, probability is high for the conflicting element to be a misspelling. Such element has little value to the system and it will be removed from KB. On the contrary, if its confidence has increased to a certain score, it will be taken as a correct place name and kept in KB.

3. Finding Synonymous Sub-tree: After removing all meaningless elements, conflicts still remain in KB between the highly confident pairs because of synonymy. In our model, the similarity between two different potential locations is computed by comparing their offsprings with preassumption that they share a similar parent. According to our experience, it is likely for different places to have sub-areas entitled with a similar names. The heuristics for determining relationship is that the more offsprings they share, the more possible they represent the same place.

4. Merging Synonymous Sub-Tree: After Resemblance is confirmed, the next step is merging two nodes together. The node with less confidence(LN) is merged into the other one(HL) and their relationship is recorded in KB as synonymy. Then, all of LN offsprings are recursively merged with HL ones. During Knowledge accumulation process, there may be some itemized place names with no direct parent due to the incomplete information. For example, in *"Fifth Avenue, Brisbane, QLD"*, the suburb is missing. We store the *"Fifth Ave"* as a new street record, whose parent identity is void. These non-parent places are referred as *dangling places* in KB.

A simple way to eliminate a dangling place is setting it as a sub-region of any place that immediately appears as its direct parent. However, such action may induce mistakes in KB considering misunderstandings by misspellings, homonymy, incorrect itemizations etc. Such mistakes can also be safely banished in KB with dangling place and its potential parent with a low confidence score at first. During the later extraction work, when the two place names appear together again, their confidence score will increase. if the score exceeds a threshold, the affiliation between these two place will be confirmed.

5 Experiments

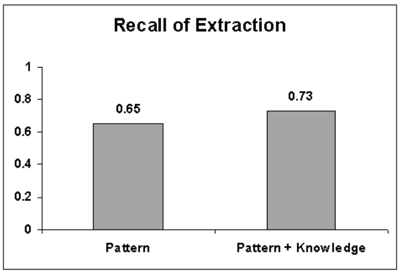
5.1 Dataset and Experimental Setup

For testing our model, we selected 1100 Web pages related to Australia. The Web pages were manually searched and addresses were extracted. Only addresses with at least street information and a recognizable location name were selected. After discarding undesirable candidates, 2030 address were chosen for the experiments.

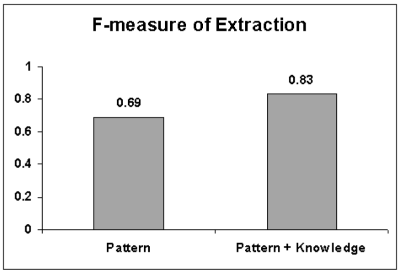
After manually control of the address, 10 most repeated patterns of addressing style in Australia were selected and the address patterns were built up with different confidence scores. We added triggers, keywords and a small set of coarse location names (country names, Australia states and major cities) with their abbreviations to our database. The database remained quite small as we basically intended to have a pattern-based approach for address extraction rather than a gazetteer approach.

5.2 Results and Discussion

The accuracy of our system has been tested in different ways. Figure 5(a) shows the recall of the proposed address extraction model. The results indicate that using address patterns alone, our system can extract 65 percent of the addresses. Adding a small table of geographic information, the recall increases to 73 percent. The results show that the pattern-based approach alone can not extract all addresses in Web pages. However, figure 6(a) shows that the precision of address extraction increases from 73 percent to 97 percent if we add a very general table of locations to the system. Figure 5(b) compares the *F-measure* of address extraction with or without using geographic knowledge. The second approach still remains more successful and this indicates that the pattern approach needs geographic knowledge to extract addresses from Web pages properly.

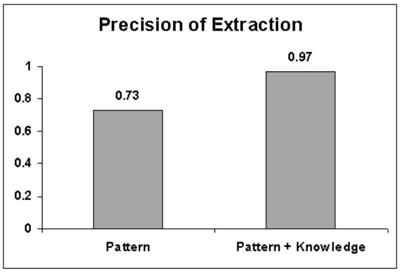


(a) Recall

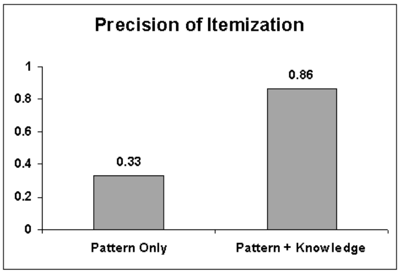


(b) F-measure

Fig. 5. Address extraction using patterns alone and using patterns and locations



(a) Address Extraction



(b) Address Itemization

Fig. 6. Precision of the system tasks: (a)Address Extraction (b)Address Itemization

For evaluating our address itemizing model, we have measured the precision of itemized addresses. Figure 6 compares the precision of address extraction and address itemization. 6(b) shows that a pure pattern-based model can not handle the itemization task properly and the precision remains poor. However, a small amount of geographic knowledge improves the itemization of addresses dramatically.

From the experimental results it could be concluded that a pure pattern-based address extraction model can not extract and itemize addresses from Web pages properly. This is because of large variation in address patterns on the Web. Finding all address patterns is almost impossible. However, as previously mentioned, adding a small table of general or coarse location names e.g. country and state names, and also using some triggers and keywords, the system provides better results both in extraction and in itemization of addresses. This combined model is cheaper and more flexible than gazetteer-based extraction approach.

6 Conclusion

In this paper, we discussed the importance of finding the addresses mentioned in Web pages as a basic source of geographic information in location-based search engines. We introduced a pattern-based address extraction approach which uses only a small table of geographic names, triggers and keywords. The results show that although a pure pattern approach can not handle all addresses properly, it can extract and itemize a reasonable proportion of addresses when a small gazetteer is used.

References

1. Amitay, E., Har'El, N., Sivan, R., Soffer, A.: Web-a-where: geotagging web content. In: SIGIR, pp. 273–280 (2004)
2. Zhou, X., Asadi, S., Chang, C.-Y., Diederich, J.: Searching the World Wide Web for Local Services and Facilities: A Review on the Patterns of Location-Based Queries. In: Fan, W., Wu, Z., Yang, J. (eds.) WAIM 2005. LNCS, vol. 3739, pp. 91–101. Springer, Heidelberg (2005)
3. Zhou, X., Asadi, S., Diederich, J., Shi, Y., Xu, J.: Calculation of Target Locations for Web Resources. In: Aberer, K., Peng, Z., Rundensteiner, E.A., Zhang, Y., Li, X. (eds.) WISE 2006. LNCS, vol. 4255, pp. 277–288. Springer, Heidelberg (2006)
4. Borkar, V.R., Deshmukh, K., Sarawagi, S.: Automatic segmentation of text into structured records. In: SIGMOD Conference, pp. 175–186 (2001)
5. Cai, D., Yu, S., Wen, J.-R., Ma, W.-Y.: Block-based web search. In: SIGIR, pp. 456–463 (2004)
6. Can, L., Qian, Z., Xiaofeng, M., Wenyin, L.: Postal address detection fromweb documents. In: WIRI '05: Proceedings of the International Workshop on Challenges in Web Information Retrieval and Integration, Washington, DC, USA, 2005, pp. 40–45. IEEE Computer Society Press, Los Alamitos (2005)
7. Chen, Y.-Y., Suel, T., Markowetz, A.: Efficient query processing in geographic web search engines. In: SIGMOD Conference, pp. 277–288 (2006)

8. Ding, J., Gravano, L., Shivakumar, N.: Computing geographical scopes of web resources. In: VLDB, pp. 545–556 (2000)
9. Etzioni, O., Cafarella, M.J., Downey, D., Kok, S., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Web-scale information extraction in knowitall (preliminary results). In: WWW, pp. 100–110 (2004)
10. Freitag, D., McCallum, A.K.: Information extraction with hmms and shrinkage. In: AAAI-99 Workshop on Machine Learning for Informatino Extraction (1999)
11. Markowetz, A., Chen, Y.-Y., Suel, T., Long, X., Seeger, B.: Design and implementation of a geographic search engine. In: WebDB, pp. 19–24 (2005)
12. Mikheev, A., Moens, M., Grover, C.: Named entity recognition without gazetteers. In: EACL, pp. 1–8 (1999)
13. Ourioupina, O.: Extracting geographical knowledge from the internet. In: International Workshop on Active Mining, ACDM-AM (2002)
14. Pouliquen, B., Steinberger, R., Ignat, C., Groeve, T.D.: Geographical information recognition and visualization in texts written in various languages. In: Handschuh, H., Hasan, M.A. (eds.) SAC 2004. LNCS, vol. 3357, pp. 1051–1058. Springer, Heidelberg (2004)
15. Sanderson, M., Kohler, J.: Analyzing geographic queries. In: SIGIR Workshop on Geographic Information Retrieval, GIR 2004 (2004)
16. Silva, M.J., Martins, B., Chaves, M., Cardoso, N.: Adding geographic scopes to web resources. In: SIGIR Workshop on Geographic Information Retrieval, GIR 2004 (2004)
17. Skounakis, M., Craven, M., Ray, S.: Hierarchical hidden markov models for information extraction. In: IJCAI, pp. 427–433 (2003)
18. Soderland, S.: Learning information extraction rules for semi-structured and free text. Machine Learning 34(1-3), 233–272 (1999)
19. Uryupina, O.: Semi-supervised learning of geographical gazetteers from the internet. In: HLT-NAACL Workshop on Analysis of Geographic References, pp. 18–25 (2003)
20. Yu, S., Cai, D., Wen, J.-R., Ma, W.-Y.: Improving pseudo-relevance feedback in web information retrieval using web page segmentation. In: WWW, pp. 11–18 (2003)