MapMarker: Extraction of Postal Addresses and Associated Information for General Web Pages

Chia-Hui Chang
National Central University, Taiwan
chia@csie.ncu.edu.tw

Shu-Ying Li
National Central University, Taiwan
965202104@cc.ncu.edu.tw

Abstract

Address information is essential for people's daily life. People often need to query addresses of unfamiliar location through Web and then use map services to mark down the location for direction purpose. Although both address information and map services are available online, they are not well combined. Users usually need to copy individual address from a Web site and paste it to another Web site with map services to locate its direction. Such copy and paste operations have to be repeated if multiple addresses are listed on a single page such as public school list or apartment list. Furthermore, associated information with individual address has to be copied and included on each marker for better comprehension.

Our research is devoted to automate the above process and make the combination an easier task for users. The main techniques applied here include postal address extraction and associated information extraction. We apply sequence labeling algorithm based on Conditional Random Fields (CRFs) to train models for address extraction. Meanwhile, using the extracted addresses as landmarks, we apply pattern mining to identify the boundaries of address blocks and extract associated information with each individual address. The experimental result shows high F-score at 91% for postal address extraction and 87% accuracy for associated information extraction.

1. Introduction

The World Wide Web contains a wealth of geographic information. People often search addresses of restaurants, hotels, schools and clubs on the Web and then use map services to mark the location. However, most of postal address information and map services are not well combined. A common task is to copy individual address from a Web page and paste it to another Web page with map services. For example, the upper part of Figure 1 shows a page of two Pittsburgh public schools with address, telephone, fax number, and other information. For people new to this city, it is necessary to have these addresses marked on a map to show the direction (see the lower part of Figure 1). Furthermore, since there are multiple schools to be marked for comparison, it is better to have additional information associated with each marker for better comprehension. Such a goal can be achieved with dozens of copy-and-paste which could be costly for busy modern people.

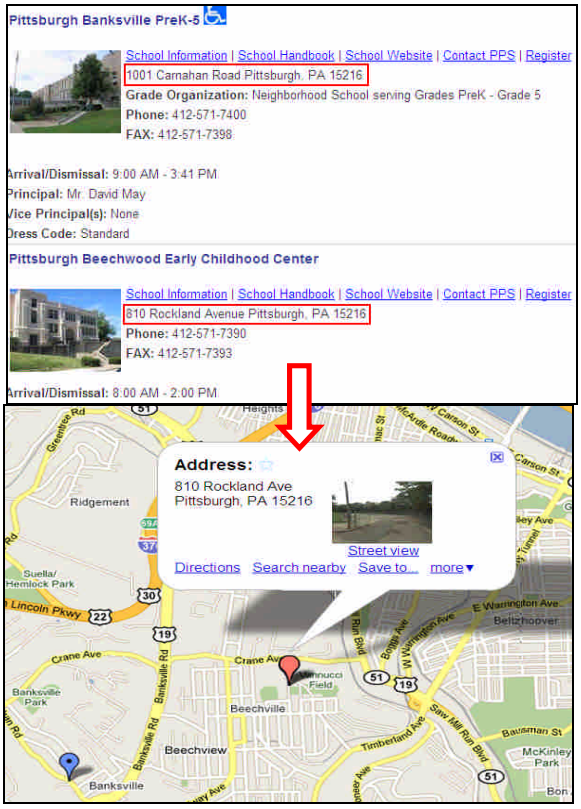

Figure 1. Postal address information and map service are not well combined.

Consider the famous Housing Mapping[1] Web site. It combines the classified ads of Craigslist[2] and Google Map[3] to create a convenience service for apartment hunting. Thus, instead of browsing though dozens of postings on craigslist, map presents a better view of the housing information searchable though location, price and number of rooms, etc. The question is whether such a service could be extended for other Web sites with postal addresses. In this paper, we endeavor to create a Web service (called MapMarker) which accepts any Web page as input and extracts any postal address with its associated information to be ready for marking on a map.

There are two modules required for such a service: one is postal address extraction, the other is associated information extraction. Both of these two problems lie within the broader area of information extraction. Postal address extraction has been studied for various countries (e.g. Australian [1], Brazil [2], etc.). In the past, address extraction basically require large gazetteers which are expensive and unavailable for many countries. Other researches apply pattern-based [1, 4, 5] or ontology-based method [2, 3, 6] to match addresses on the Web page. In this paper, we apply machine learning based approach for address extraction and obtain an increase in accuracy from 0.876 to 0.914 (F-score) compared with Yu [9].

Associated information is intended to disambiguate the issue caused by multiple addresses on the map. Using extracted addresses as landmarks, the second problem is to extract associated information with each address. For web pages that contain only one postal address, it is sufficient to use page title or named entity mentioned in the Web page as additional information. However, for web pages that contain multiple addresses, the task is more challenging. In this paper, we apply pattern mining to identify the boundaries of address blocks and extract associated information with each individual address. The experimental result shows 87% accuracy for associated information extraction.

2. Related Work

Previous researches on information extraction from documents have been studied for many years. Many conferences such as Message Understanding Conference (MUC) and Text Retrieval Conference (TREC) have invested lots of efforts in the field of information extraction. Address is a specific information type that is very useful in our daily life. We briefly review recent researches by their extracting methods: pattern-based, ontology-based and machine learning method.

Pattern-based method [1, 4, 5] often relies on Gazetteer to provide names for states, cities and streets. A Gazetteer is a geographical dictionary, an important reference for information about locations and place names. Saeid et al [1] introduce a pattern-based address extraction approach which uses several address patterns and a small gazetteer. Both HTML and visual-based segmentations are used to increase the quality of address extraction. The F-score is 0.83 and the recall is 0.73. Lin et al [4] use vision-based text segmentation based on layout formatting information. Then they apply address patterns to identify if a text block is a postal address. The result shows that the approach with a high precision 0.89. However, the testing data set includes 44 web pages.

Ontology is a formal representation of a set of concepts within a domain and the relationship between those concepts. Cai et al [6] employ an ontology-based conceptual information retrieval approach combined with graph matching techniques to automatically identify possible address structures in a Web page. Experimental evaluation shows that the method yields F-score 0.734 and the recall 0.724 since graph matching is a difficult problem. Karla et al [2, 3] presents an ontology-based approach that helps recognize and extract geospatial evidence with local characteristics, such as street names and area codes. The experimental result shows that the F-score is about 0.77.

In recent years, there are some researches who apply machine learning technique for address extraction. Ourioupina et al [7] describe an algorithm to classify places into six location types (city, region, country, island, river and mountain) and determine for a given place name, where the place is. They apply C4.5 Decision Tree and 10-fold cross validation to train and test data. The accuracy is about to 89%. Uryupina et al [8] present an approach to the acquisition of geographical gazetteer instead of creating these resources manually. The Bootstrapping approach is investigated to create new gazetteers using only a small seed dataset. The system produces six binary classifiers and determines a class (CITY, ISLAND, RIVER, MOUNTAIN, REGION and COUNTRY) of any geographical name. Classifiers perform with the average accuracy of 0.86.

In Yu‘s research [9], he proposes a hybrid method which combines the pattern-based and machine

[1] Housing Mapping : http://www.housingmaps.com/
[2] Craigslist: http://www.craigslist.org/about/sites
[3] Google Map: http://maps.google.com/

learning approach. The system would pre-process the web pages into tokens and output them in sequence with each token labeled as one of four classes (START, MIDDLE, END and OTHER). At the final step, post-processing will be applied to the labeled tokens to extract and output addresses. More than 1700 web pages are used as dataset. The experimental evaluation shows that F-score is 0.876 and the recall is 0.811.

The rest of the paper is organized as follows. Section 3 describes the address extraction module based on conditional random field. Section 4 presents the algorithm for associated information extraction. The experimental results are shown in section 5. Finally, conclusion and future work are given in Section 6.

3. Address Extraction

Generally speaking, an address is composed of street number, street names, city names, province, country names and ZIP code. Some components, like ZIP code and directions, are optional and some components can be placed in different order. In the past, address extraction techniques basically require large gazetteers which are expensive and unavailable for many countries. Recently, some studies have applied structured learning based on decision tree or SVM to train model for extracting addresses. Since the techniques of unstructured learning based on HMM or CRFs have progressed tremendously, we adopt unstructured learning based on sequence labeling to solve this problem.

Our address extraction module composes of three main modules: pre-processing, feature extraction and learning module. As shown in Figure 3, pre-processing will be conducted first to find out candidate segments. A set of features are then identified for the learning module to adjust weights for labeling the beginning and ending of addresses in the candidate segments. Finally, we can obtain extracted addresses from testing pages through a similar procedure and the learning model. Let us discuss the following modules in full detail.

3.1. Pre-processing

Pre-processing can be divided into three steps: ANNIE annotation, tokenization and segmentation. ANNIE system is a mature Name Entity Recognition (NER) which is plugged in GATE system. It is implemented for labeling different kinds of name entities in many documents. We adopt ANNIE system to annotate geographical names and positions of the Web page.

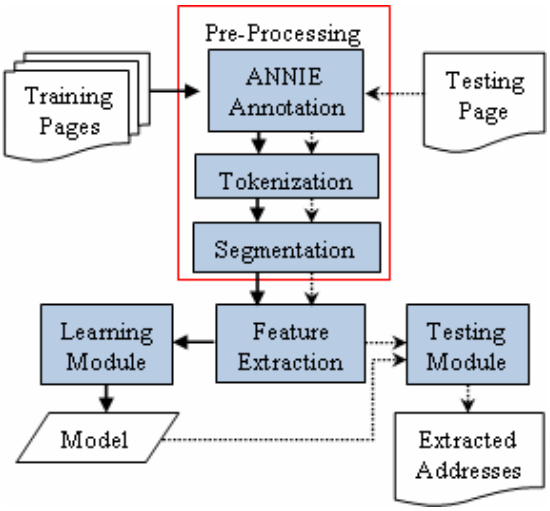

Figure 2. Flow chart of address extraction

Using the annotated page, tokenization is applied to eliminate HTML tags and split content of Web page into a sequence of tokens based on the blanks and punctuations.

The third step is segmentation of the token string based on address keyword matching. Table 1 shows four kinds of address keywords and related segment criteria. Address keywords include street suffix, state names, directions, country names and their abbreviations. Given a matched address keyword, we will cut a segment from *m* tokens before the matched address keyword until the *n* tokens after the matched address keyword. That is, a total of *m+1+n* tokens are merged as a candidate segment.

Table 1. Address keywords and criteria

Type	Segmenting criteria		Example
	m	n	
Street suffix	6	2	Street, St.
State name	2	4	PA, Florida
Direction	2	2	North, N.,
Country name	2	2	USA, Canada,

If two candidate segments overlap, we combine the overlapped segments to form a longer segment. These segments are called candidate segments for possible postal addresses.

3.2. Feature Representation and Labels

For each token, we enumerate fourteen features that could be indications of an address segment. For example, whether a token denotes a state name, direction, street /avenue abbreviation, zip code are the essential elements for being an address segment. In addition, token of telephone number, some punctuation like colon, could be a good sign for nearby address segments. Other characteristics such as whether a token is all capitalized, initially capitalized, or all digitals could denote an abbreviations of countries, city names, or street number, etc. Table 2 lists the description of each feature and corresponding examples of the 14 features used in this paper.

Table 2. Feature description and examples

	Feature	Examples
1	ALLCAPS	CA, USA
2	ALLLOWER	suite
3	INITIALCAPS	Street, Road
4	ALLDIGITS	95846
5	PUNCTUATION	, - :
6	DIGITPUN	12th
7	INITIAL	N.
8	LETTERPUN	Address :
9	STREET	St., Avenue, Ave
10	STATE	California, CA
11	DIRECTION	North, NW
12	PHONE	(925)223-25459
13	ZIPCODE	96548, 96584-1547
14	ANNIE Location	True / False

Like many information extraction tasks, we also use BIEO tagging as the labels of each token: where B stands for the beginning position of a postal address, I stands for the inside position of a postal address, E stands for the ending position of a postal address, while O stands for outside a postal address. Figure 3 shows an example of the BIEO tagging format. Once a testing sequence has each token labeled with proper labels, we then extract a segment with a beginning B tag, ending E tag, and at least two inner I tags.

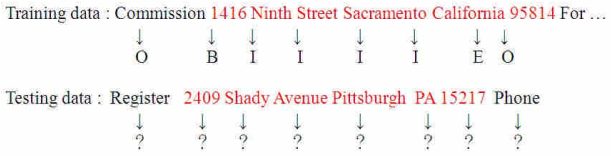

Figure 3. Information extraction as a sequence labeling (BIEO) task

3.3. Learning Models

We use two learning models for the labeling task: one is support vector machine and the other is conditional random fields. The former is structured learning and the latter is unstructured learning. We adopt these two approaches as our learning models because they are best known discriminative models for structure and unstructured learning, respectively. SVM is introduced by Boser [13]. The basic concept of SVM is to find a hyper-plane to separate two sets of data apart with maximum margin and minimum misplacement. Formally, SVM solves the following constraint optimization problem.

$$\min_{\mathbf{w},b,\xi} \arg \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i$$

$$y_i\left(\mathbf{w}^T\phi(\mathbf{x}_i) + b\right) \geq 1 - \xi_i, \xi_i \geq 0, i = 1,..., N$$

where $\boldsymbol{w}$ and b are the coefficient for the hyper plane, ϕ is the kernel function, C >0 is the penalty parameter and ξ_i are the slack variables for mislabeled examples. SVM has successful applications in many fields such as bioinformatics, text and image recognition. In this paper, we apply libsvm [] with polynomial kernel.

To decide the label of each token, we include the features of current token as well as those of previous and following tokens. A window size of 2n-1 (n=2) tokens is used to train the model (see Figure 4).

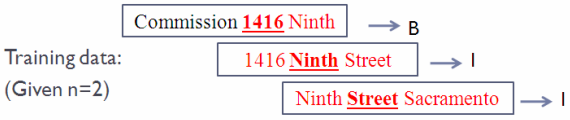

Figure 4. Structured learning using sliding window

Conditional random fields [10, 12] are undirected graphical models which define conditional probability distribution over labeled sequences given a particular observation. Compared to generative models such as HMM, CRF has the advantage of relaxing the independence assumptions required in calculating the joint probability. In addition, CRF also avoid the bias toward states with fewer successor states in other directed graphical models like MEMM. CRF are now widely used for sequence segmentation and labeling task in many fields such as bioinformatics, computational linguistics, speech recognition, etc.

Let X be a random variable over the observations and Y_i be a random variable over corresponding label alphabet Ψ. Let G=(V,E) denote the undirected graph (the shaded nodes and edges in Figure 5) such that nodes denotes the random variables $Y=\{Y_v\}_{v\in V}$. and edges denotes dependency between them. Then (X,Y)

is said to be a conditional random field if, when conditioned on X, the random variables Y_v obey the Markov property with respect to the graph: $p(Y_v|X,Y_{V-\{v\}})=p(Y_v|X,Y_{Nv})$ where Y_{Nv} denotes the set of neighbors of the node v in graph G.

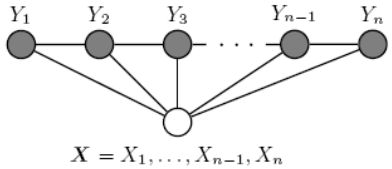

Figure 5. Linear-chain CRF with random variables Y and observation X.

Formally, the conditional distribution of the labels Y given the observations X has the form,

$$p(y \mid x) = \frac{1}{Z(x)} \exp\left(\sum_{e \in E,k} \lambda_k f_k(e, y_e, x) + \sum_{v \in V,k} \mu_k g_k(v, y|_v, x) \right)$$

where f_k and g_k are edge and node feature functions, respectively ($y|_e$ and $y|_v$ are nodes associated with edge e and vertex v), λ_k and μ_k are parameters to be estimated from the training data and Z(X) is the normalization factor, which has the form,

$$Z(x) = \sum_{y} \exp\left(\sum_{e \in E,k} \lambda_k f_k(e, y|_e, x) + \sum_{v \in V,k} \mu_k g_k(v, y|_v, x) \right)$$

There are several packages available for CRF. In this paper, we adopt CRF++ which is an open source implementation based on LBFGS, a quasi-newton algorithm for large scale numerical optimization problem. The input file to CRF++ consists of token sequences, where each token is represented by the 14 features described in Section 3.2. Empty lines are used to identify segment boundaries.

In addition to the 14 primal features for each token, CRF++ also provides feature templates and macro that can generate different composite features. There are two types of templates: Unigram and Bigram output labels. The former make use of current output label with primal features coded by macro to generate different feature functions. While the later also uses previous output labels, resulting more feature functions. In this task, both types are combined and applied to generate a great deal of different features.

4. Associated Information Extraction

Let us now shift the emphasis from address extraction to associated information extraction. The major purpose of this task was to provide users associated information with individual postal addresses for better comprehension. While considerable attention has been paid in the past to research issues related to information extraction, supervised learning approaches are often limited by specific information. For example, apartment renting requires bed room information, while job hunting entails individual ability description. There would be too many kinds of information to be extracted. On the other hand, unsupervised approaches based on pattern discovery could overlook such details and provide general information associated with the extracted address. For the above reasons, the secondary purpose of this task is to propose a suitable and robust method to extract associated information for web pages with different templates.

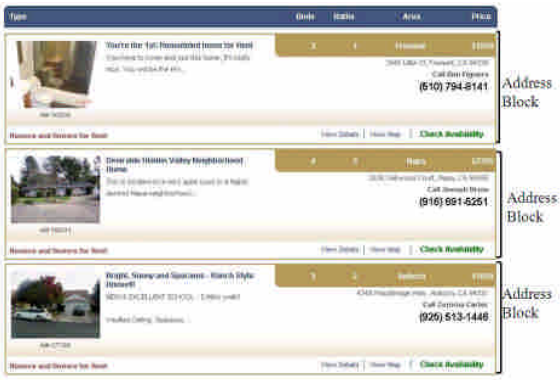

Figure 6. Address blocks.

As shown in Figure 6, we observe that many web pages with multiple postal addresses often have associated information forming address blocks and arranged regularly. Since the rendering is controlled by HTML tags, the HTML tags in the web page should present some regularity. Based on these observations, the issue becomes how to find the boundaries among the sequence of HTML tags.

Given the extracted address a_1, a_2, …, a_k from the web page P, the algorithm first locates the start position of each extracted address a_i in P (Step 1), then match backward (s_i=s_i-1) the tags from s_i-1 until any mismatch occurs, where we set the last common tag as the separator of the address block (Step 2). Meanwhile, the position s_i is recorded as the beginning of the address block.

Next, we calculate the average gap (avgG) between two adjacent address blocks. We remove gaps that are greater than 1.5 avgG and re-calculate the average gap to estimate the end position (e_i=s_i+avgG) of address blocks (Step 3). We then match backward to find the separator tag discovered above (Step 4). The idea behind this recalculation of average gap is to avoid

large gaps that are caused by missed address. As shown in Figure 8, if address 4 is not extracted due to mislabeling in the learning model, the algorithm will remove *g3* to give a closer estimation of the average length for end position. Finally, the segment between s_i and e_i is used to denote the associated information we need.

Input: Web page P and extracted addresses $\{a_1,a_2,\ldots, a_k\}$
Output: Address block $\{B_1,B_2,\ldots,B_k\}$, where $B_i=(s_i, e_i)$, for i = 1 to k

1. Find the start position s_i of each extracted addresses a_i (i = 1 to k) in the web page p.
2. Match backward tags of s_i (i = 1 to k) until mismatch occurs
   Repeat
      $s_i = s_i - 1$; for all i
   Until $Tag(s_i) \neq Tag(s_j)$ for some i and j
   $s_i = s_i + 1$; for all i
   Set separator = $Tag(s_1 + 1)$
3. Find average length among the separators
   Set $g_i = s_{i+1} - s_i$; for all i
   Set $S= \{g_1\} \cup \{ g_2 \} \cup \ldots \cup \{ g_{k-1} \}$
   avgG = average(S) .
   Delete g_i from S if $g_i > 1.5$ *avgG, for all i
   avgG = average(S).
4. Find end positions for each B_i (i= 1 to k)
   Set $e_i = s_i + \lceil avgG \rceil$; for all i
   Repeat
      $e_i=e_i - 1$; for all i
   Until $Tag(e_i)$ = separator
5. Adjusting separator to ensure non-overlapping
   Sum += e_i-s_i; for all i
   If Sum > e_k-s_1
      $s_i = s_i + 1$; for all i
      Set separator = $Tag(s_1 + 1)$
      Go to step 3
   Endif

Figure 7. Algorithm for Associated information extraction

The final step of the algorithm adjusts the separator to ensure non-overlapping of address block (Step 5). This prevents the case when the first address in a list is not identified by the address extraction step or any extension of the pattern to non address area, resulting incorrect starting position and separator. The detection is done by comparing the summation of gaps between adjacent addresses ($\sum_i e_i-s_i$) with the gap between the starting position of the first address and the end position of the last address (e_k-s_1). If the former ($\sum_i e_i-s_i$) is greater than the later (e_k-s_1), we try next common separator ($s_i = s_i + 1$) and repeat step 3, 4 and 5 until the condition is released. This adjustment can deal with mistakes which occurs at the first two steps.

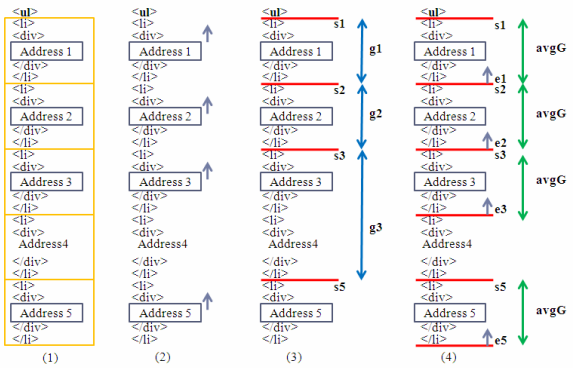

Figure 8. Example of associated information extraction

5. Experimental Results

5.1 Address Extraction

We use dataset from Yu, et al [9] paper as our training and testing data. The dataset is obtained by three topic queries (Contact, Hotel and Pizza) from Google. Among them, 1205 pages are with single address and 535 pages are multiple addresses pages (with 8519 postal address). The performance is measured in terms of label accuracy and extraction accuracy. All reported values are the results of 10 fold cross validation. Table 3 and 4 shows the F-score of B, I, E labels and the F-score of postal addresses, respectively.

Table 3. F-score of B, I and E class.

Target Label / Method	B class	I class	E class
Regular Expression	0.677	0.848	0.875
Gazetteer	0.777	0.858	0.838
C4.5 Decision Tree	0.873	0.928	0.906
SVM	0.878	0.993	0.987
CRF	0.927	0.993	0.959

The first three rows show the performance of regular expression, gazetteer, and C4.5 decision tree obtained from Yu [9]. Note that regular expression has lowest performance since it cannot cover all possible formats for postal addresses. Gazetteer is higher than regular expression since it contains lots of geographical information for matching and extracting. C4.5 decision tree combined with word n-gram model has the best performance 0.876 among them. The experimental result shows that the proposed approach outperform existing methods by 0.5~6 percentage. As B, I and E are the critical labels for address extraction, the performance of the proposed approaches also have

increased result for postal address extraction as shown in Table 4. The F-score for SVM is 0.903 with an increase of 2.7%, while the F-score for CRF has the best result of 0.914 with an increase of 3.8%.

Table 4. Comparison of address extraction.

Method / Performance	Precision	Recall	F-score
Regular Expression	0.831	0.607	0.753
Gazetteer	0.735	0.689	0.665
C4.5 Decision Tree	0.952	0.811	0.876
SVM	0.961	0.851	0.903
CRF	0.968	0.865	0.914

Finally, we compare the effect of including ANNIE in the preprocessing step. Table 5 shows four combination of SVM and CRF, with or without ANNIE's tagging information. As we can see, even without ANNIE, SVM and CRF still outperform C4.5 with an increase in F-score from 1.8%~3.3%, showing that SVM and CRF have better learning capability than C4.5. On the other hand, ANNIE has accounted for the remaining increase (around 0.9%~0.5% in F-score).

Table 5. Effect of using ANNIE.

Method / Performance	Precision	Recall	F-score
SVM without ANNIE	0.959	0.837	0.894
SVM with ANNIE	0.961	0.852	0.903
CRF without ANNIE	0.963	0.861	0.909
CRF with ANNIE	0.968	0.866	0.914

5.2 Associated Information Extraction

Next, we show the performance of associated information extraction. Of the 535 web pages which contain multiple addresses, 8041 addresses are extracted from 8519 addresses in these pages. Based on these extracted addresses, we extract possible address block with the approach proposed in Section 4. Table 6 summarizes the accuracy for associated information extraction with or without adjustment (Step 5). The experimental result shows that the accuracy is 0.851 without adjustment and improved about 1.5% with adjustment.

Table 6. Performance of associated information extraction

	Correct Boundary	Wrong Boundary	Accuracy
Before adjustment	6840	1201	0.851
After adjustment	6987	1054	0.869

We analyze the reasons why extracting address blocks fails as follows. There are two possible cases: (1) address blocks have more than one layout for displaying postal addresses (e.g. Figure 9). (2) a web page may not have obvious boundaries among addresses because this page have addresses and description intertwined without common template (e.g. Figure 10). Although we propose adjusting method, it is still hard to recover complete information correctly.

Figure 9. Two different layouts in a web page.

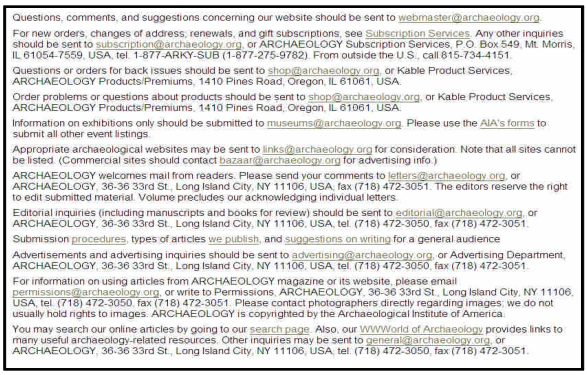

Figure 10. No obvious boundary among extracted addresses.

6. Conclusions and Future Works

In this paper, we propose the service of extracting postal addresses from general Web pages and combine with map service to provide visualization and comprehension of address information. The service is implemented with two information extraction tasks: postal address extraction and associated information

extraction. We applied two discriminative learning models: SVM and CRFs to train the learning model and test dataset. The experimental results show that the F-score of address extraction is improved from 0.876 (Yu [9]) to 0.903 and 0.914 based on SVM and CRF, respectively. Based on extracted addresses, we further discover the separators among address blocks to extract associated information from original web pages. The accuracy of associated information extraction is around 0.869. With the extracted addresses and associated information, we can mark them on Google Maps to provide visualization for postal address information. A demo web site can be accessed at http://140.115.51.19/map/map.html (See Figure 11).

For future work, associated information still has a big gap to be improved. Finding separator in the forward direction or deploy with bi-direction search could further improve the extraction performance. In addition to improve the extraction performance, we also prepare to extend the extraction to other languages and deploy such techniques for location based searches.

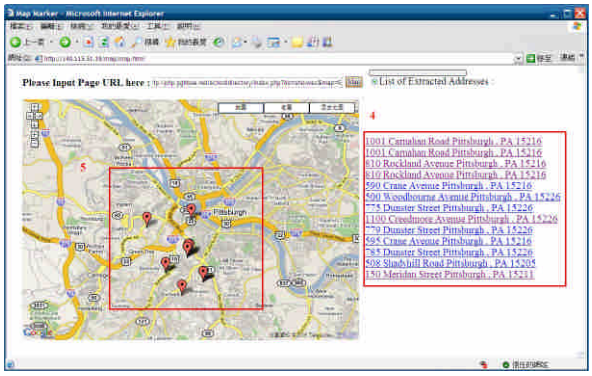

Figure 11 Demonstration of MapMarker service

References

[1] S. Asadi, G. Yang, X. Zhou, Y. Shi, B. Zhai, W. Jiang: Pattern-Based Extraction of Addresses from Web Page Content. APWeb 2008: 407-418.

[2] K.A.V. Borges, A.H.F. Laender, C.B. Medeiros, C.A. Davis: Discovering geographic locations in web pages using urban addresses. GIR 2007: 31-36.

[3] K.A.V. Borges. Use of an Ontology of Urban Places for Recognition and Extraction of Geospatial Evidences on the Web (in Portuguese). PhD Thesis, Federal University of Minas Gerais: Belo Horizonte (MG), Brazil, 2006.

[4] L. Can, Z. Qian, X. Meng, W. Lin: Postal Address Detection from Web Documents. WIRI 2005: 40-45.

[5] P. Nagabhushan, S.A. Angadi, B.S. Anami: A Fuzzy Symbolic Inference System for Postal Address Component Extraction and Labelling. FSKD 2006: 937-946.

[6] W. Cai, S. Wang, Q. Jiang: Address Extraction: Extraction of Location-Based Information from the Web. APWeb 2005: 925-937.

[7] O. Ourioupina. 2002. Extracting geographical knowledge from the internet. In Proceedings of the ICDMAM International Workshop on Active Mining.

[8] O. Uryupina. (2003) Semi-supervised learning of geographical gazetteers from the internet. In: Kornai, A. and Sundheim, B. (eds.) Proceedings of the HLT-NAACL 2003 Workshop on Analysis of Geographic References, Alberta, Canada: ACL, 18-25.

[9] Z. Yu. High Accuracy Postal Address Extraction From Web Pages. Dalhousie University. 2007.

[10] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. Proc. 18th International Conf. on Machine Learning, 2001.

[11] R. Klinger and K. Tomanek. Classical Probabilistic Models and Conditional Random Fields. Algorithm Engineering Report TR07-2-013, Department of Computer Science, Dortmund University of Technology, December 2007. ISSN 1864-4503.

[12] C. Sutton and A. McCallum. An Introduction to Conditional Random Fields for Relational Learning. In "Introduction to Statistical Relational Learning." Edited by Lise Getoor and Ben Taskar. MIT Press, 2006.

[13] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers", Proceedings of the Fifth Annual Workshop on Computational Learning, 1992.