3M2-IOS-3b-3

On Chinese Postal Address and Associated Information Extraction

Chia-Hui Chang, Chia-Yi Huang and Yueng-Sheng Su

National Central University, Taiwan

Address information is closely linked to people's daily life. People often need to query addresses of hotels, schools, and organization of new environment; and use map service to mark the real location. MapMarker is a service, which extracts English postal addresses from general web pages and marks them on the map with associated information. This paper extends the idea to Chinese postal addresses extraction and improves the extraction of associated information for each address with hierarchical clustering. We show how to apply sequence labeling technique on Chinese postal address extraction using both BIEO and IO tagging methods. We compare the performance with and without Yahoo Chinese word segmentation. The results show that Chinese postal addresses can be extracted through conditional random field (CRF) with high F-measure 0.97 using BIEO tagging without word segmentation since incorrect segmentation can lead to worse labeling of address tokens. Meanwhile, associated information for each address is also identified based on clustering of the addresses into address blocks. The F-measure is improved from 0.90 to 0.92.

1. Introduction

Address is the information that is constantly used in our daily life. For instance, when traveling for business or leisure, tourists may need to locate hotels near the scenic spots or transport stations; people who search for a job may need to acquire the company address; drivers with a broken car may need to find a parking lot. Although the Web contains a wealth of geographic information, most 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 other Web site with map service. Thus, a necessity here is to combine address information with map service. For example, Housing Maps (http://www.housingmaps.com/) is a famous mashup that combines the classified ads of Craigslist (http://www.craigslist.org/) and Google Map (http://maps.google.com/) to create a convenience service for apartment hunting. Therefore, Chang and Li [6] proposed the idea of MapMarker which accepts any Web page as input and extracts postal addresses with their 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 [2], Brazil [3], etc.). In the past, address extraction basically require large gazetteers which are expensive and unavailable for many countries. Other researches apply pattern-based [2, 4, 9] or ontology-based method [3, 5] to match addresses on the Web page. Associated information extraction is intended to alleviate the confusion when multiple addresses are placed on the map such that whenever a mark on the map is clicked, associated information of that address can be displayed. For web pages that contain only one postal address, it is often sufficient to use page title or named entity mentioned in the Web page as associated information for disambiguation. However, for web pages that contain multiple addresses, the task is more challenging.

Contact: Chia-Hui Chang, No. 300 Jhongda Rd., Jhongli City, Taiwan, Fax: +886-3-4222681, E-mail: chia@csie.ncu.edu.tw

In this paper, we enhance the MapMarker service to Chinese postal address extraction and improve associated information extraction as well. Since there is no delimiters between word boundaries in Chinese, many Chinese nature language processing would require Word Segmentation as a preprocessing step.

The rest of the paper is organized as follows. Section 2 describes related work on address and information extraction. Section 3 introduces our methods for Chinese address extraction. Section 4 illustrates the method for associated information extraction. The experiment is explained in Section 5. Finally, we discuss and analyze experiment result for conclusion.

2. Related work

Information extraction (IE) is the task of automatically extracting structured information from unstructured text documents, which is designed to test how much machine can understand the messages written in natural language and automate mundane tasks performed by human. One of the major conferences on IE is DARPA sponsored Message Understanding Conference (MUC) from 1987 to 1998. IE is also important for semantic Web which intents to allow the data to be shared and reused across applications. Due to the difficulty of the problem, current approaches to IE focus on narrowly restricted domain. Address extraction is a specific kind of information extraction, which intends to extract complete addresses and has many social and economical applications. In addition to complete postal address extraction, the task of address component labeling (also called address normalization) is to segregate different components under different labels is also important [6].

Generally speaking, a complete address can be composed of housing number, street, city, region, county, state, zip code, etc. To indentify the address boundary, an IE program often uses keywords such as “road”, “street” or “avenue” to locate candidate strings that might contain a complete address, and then determine the boundary using various approaches. In this paper, we briefly review recent researches by their extracting methods: pattern-based and machine learning method.

2.1 Pattern-Based Methods

Pattern-based methods generally rely on Gazetteer or ontology to provide names for cities and states. Gazetteer is a geographical dictionary, which is an important reference for location names, while ontology is a formal representation of a set of concepts within a domain and the relationship between those concepts.

Asadi et al. [2] proposed a pattern-based address extraction approach that uses several address patterns and small geography knowledge to extract English postal addresses. They also adopted HTML and vision-based segmentation to improve overall extraction quality with an F-measure to 0.83. Besides, Lin et al. [8] used layout format to divide input pages into text blocks and verify whether a candidate text block contains a postal address with address patterns. This method has an accuracy of 0.89. Nevertheless, 44 pages for testing are not quite enough.

Cai et al. [5] designed an address ontology with suite information, municipal location, region position and applied graph matching technique for English address extraction. The result shows an F-measure of 0.734 on 6 yellow pages and 5 Yahoo business pages. Borges et al. [3] proposed 11 patterns based on an ontology with basic address, complement, location identifier and subcategories like phone number, zip codes, etc. An extraction of Brazilian addresses from a collection of over 4 million Web pages was conducted. However, precision and recall are not reported due to a large number of testing data.

2.2 Learning-based methods

Supervised learning methods are widely used in many applications. There are two kinds of supervised learning tasks including classification and sequence labeling. Classifier like Naïve Bayes, C4.5 Decision Tree or Support Vector Machine (SVM) are commonly used learning algorithms which assign a label to a given input, while Hidden Markov Model (HMM), Maximum-Entropy Markov Model (MEMM) and Conditional Random Fields (CRF) are designed for assigning a label to each member of a sequence of inputs.

Ourioupina et al. [10] defined a classification problem with six location types (including city, region, country, island, river and mountain). An accuracy of 89.1% is reported with C4.5 decision tree classifier and 10-fold cross validation. Uryupina et al. [14] extended this framework to create new gazetteers with a small dataset through bootstrapping approach.

Yu [15] proposed a hybrid method which combined pattern-based and machine learning approaches by classify each token into four classes: START, MIDDLE, END, OTHER. They then segmented the string from START label to END label as a complete address. On a dataset of more than 1700 pages, they showed an F-measure with 0.876 using C4.5 decision tree.

Recently, Chang and Li et al. [6] applied CRF with BIEO tagging (BEGIN, INTERMENIATE, END, OTHER) for English address extraction. With 14 designed features, the performance can be improved to 0.913 F-measure on Yu's dataset. Meanwhile, associated information extraction is another important module which needs improvement. In this paper, we shall focus on Chinese address extraction and improve associated information extraction as well.

3. Chinese Address extraction

We can view address extraction as a sequential labeling problem. Our approach for Chinese address extraction is composed of three steps. As shown in Figure 1, the preprocessing step includes two modules, namely tokenization and candidate string segmentation. Then, a set of features are calculated for the learning module to adjust weights for maximizing log likelihood. At testing, the input page will go through the preprocessing step and feature extraction. Then, the trained model is used to label candidate strings to obtain a complete address.

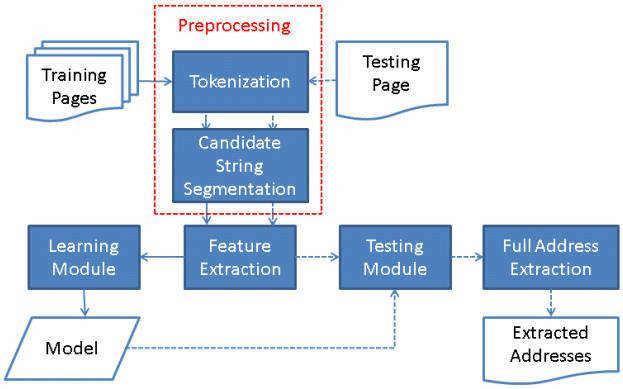

Figure 1: Flowchart of Chinese address extraction

3.1 Tokenization

The preprocessing step can be divided into two parts: tokenization and candidate string segmentation. Unlike English, where words are delimited by spaces and commas, Chinese sentences are sequences of Chinese characters where each word is made up from 1 to 4 characters. Therefore, word segmentation has been an important step for Chinese language processing. However, it is also possible to use individual Chinese characters as the basic units for processing. In this paper, we use Yahoo Chinese Word Segmentation (YCWS) service as well as individual characters for tokenization.

- Yahoo Chinese Word Segmentation (YCWS): YCWS is a standard application interface provided by Yahoo for Chinese word segmentation. It integrates linguistics and information retrieval with enormous corpus in Yahoo. For instance, a string "桃園縣中壢市中大路 300 號" will be divided into 7 tokens: "桃園", "縣", "中壢", "市", "中大路", "300", "號".
- Individual Chinese Characters Segmentation (ICCS): For this tokenization, each individual Chinese character is considered a token. For example, "桃園縣中壢市中大路 300 號" will be divided into 11 tokens: "桃", "園", "縣", "中", "壢", "市", "中", "大", "路", "300", "號".

3.2 Candidate String Segmentation

To locate candidate strings that might contain address, we use a set of keyword suffixes as landmarks and extend additional tokens to form a candidate string. Suppose the extended window is *w* tokens, the system would include *w* tokens before and after each key character to generate a segment of *2w+1* tokens. If two

candidate strings are overlapped, we connect them as one candidate string.

Suffix Types	Key Characters
Administrative division	縣(county),市(city),鎮(town), 鄉(township),區(district), 村 (village), 里 (neighborhood), 鄰 (neighbor)
Street	道(avenue),路(road),街(street),段(section), 巷(lane), 弄(alley)
House No.	號(number), 樓(floor), 室(room)

Table 1: Common Chinese address suffixes.

Figure 2 shows three suffix types that are used as landmarks for candidate string segmentation. For example, administrative division suffixes include 縣 (county), 市 (city), 鎮 (town), 鄉 (township), 區 (district), 村 (village), 里 (neighborhood), 鄰 (neighbor); street suffixes include 道 (avenue), 路 (road), 街 (street), 段(section), 巷(lane), 弄(alley); while house numbers include 號(number), 樓(floor), 室(room). As an example, the keyword "縣" (county) will segment a string of "學堂地址: 510 彰化縣員林鎮山腳路三段" from Figure 2. Similarly, the keyword "鎮" (town) will segment a string of "址: 510 彰化縣員林鎮山腳路三段 2 巷 6". Since these two candidate segments are overlapped, it will be connected as one candidate segments. The process is repeated to deal with other keywords, such as "鎮", "路", "號". The parameter *w* of extended length will be determined by experiments with corresponding tokenization.

Figure 2: Candidate String Segmentation

3.3 Feature Extraction

For each token, we design seventeen features that are used as indicators of an address segment. For character tokens, we use key characters from the three suffix types and consider additional compositions of Chinese addresses. For example, three features based on area size are used for administrative divisions (namely, county/city, township, and village). We assign one to those features if the tokens are the corresponding key characters. Similarly, two features are used for street (street/road and lane/alley) and house number (house number and building). As for word tokens, we make use of Taiwan's 368 counties from Wikipedia (153 townships, 41 towns, 17 county-controlled cities, and 157 districts) as the gazetteer. For remaining features, a word token will be assigned a value of one if the word token has street/road suffix characters. Thus, no separate gazetteer is used.

In addition, we also have contact tags like "telephone" (電話) and "address" (地址) as one of the features since these tokens come along with Chinese addresses frequently. On the other hand, zip codes and room numbers are usually numbers with fixed length, while street numbers often have variable length. As a result, we choose six lengths of numbers from length 1 to 5 as well as extra long and assign value 1 to one of these six features. Table 2 enumerates the features designed for ICCS and YCWS tokenization.

	Feature	ICCS	YCWS
1	CountyCity	縣, 市	台中縣, 台北市
2	Township	鎮, 鄉, 區,	蘆竹鄉, 大安區
3	Village	村, 里, 鄰	三合村, 五分里
4	StreetRoad	道, 路, 街	八德路, 和平街
5	LaneAlley	段, 巷, 弄	三巷, 九弄
6	HouseNo	號	十五號
7	Building	樓, 室	六樓, B 室
8	ContactTag	地, 址, 電, 話	地址, 電話
9	Punctuation	、：；，・:;., ""	
10	ChineseNo	一, 二, 三, 四, 五, 六	
11	AllDigits	42011, 0937137659	
12	DigitLen1	5, 6, 7	
13	DigitLen2	11, 32	
14	DigitLen3	420, 260	
15	DigitLen4	5566, 1234	
16	DigitLen5	42011	
17	DigitLong	327363, 4227151	

Table 2: Feature selection

3.4 Learning module

After preprocessing and feature extraction, we train a model with the learning module. In the research, we use conditional random fields (CRF) as the learning algorithm.

(1) Conditional Random Fields

Conditional random fields [7, 13] 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.

A special kind of CRFs, one where all the nodes in the graph form a linear chain, is commonly used for information extraction. The problem of learning a linear-chain CRF can be defined as follows: Given a sample set of training sequences $\{X_1, X_2, \dots\}$ along with their corresponding label sequences $\{Y_1, Y_2, \dots\}$. The task of learning is to find the best possible potential functions such that the log-likelihood is maximized. Once the potential functions are determined, the inference process can be defined: Given a new observable sequence x, find the most likely label sequence Y* for x, i.e. compute $Y^*=\text{argmax}_Y P(Y|X)$.

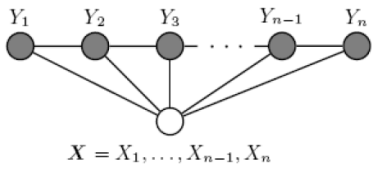

Figure 3: Linear-chain CRF with random variables Y and observation X.

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

$$P(Y_1,...,Y_N \mid X) = \frac{1}{Z_X}\prod_{i}^{N}\psi_i^v(Y_i,X)\psi_i^e(Y_i,Y_{i-1},X) \quad (1)$$

where $\psi_i^v(Y_i,X)$ and $\psi_i^e(Y_i,Y_{i-1},X)$ are potential functions for node and edge, respectively. Most sequence labeling applications use log-linear potential functions.

$$\psi_i^v(Y_i,X) = \exp\left(\sum_j \mu_j g_j(Y_i,X,i)\right) \quad (2)$$

$$\psi_i^e(Y_i,Y_{i-1},X) = \exp\left(\sum_k \lambda_k f_k(Y_i,Y_{i-1}X,i)\right) \quad (3)$$

where g_j and f_k are node and edge feature functions that depends on the labels at current position and previous position, μ_j and λ_k are parameters to be estimated from the training data and Z_X is the normalization factor, which has the form,

$$Z_X = \sum_Y \prod_i^N \psi_i^v(Y_i,X)\psi_i^e(Y_i,Y_{i-1},X). \quad (4)$$

There are several packages available for linear chain CRF. In this paper, we adopt CRF++ [1] which is an open source implementation based on LBFGS, a qusi-Newton algorithm for large scale numerical optimization problem. The input file to CRF++ consists of token sequences, where each token is represented by the 17 features described in Section 3.3. Empty lines are used to identify sequence boundaries.

In addition to the 17 primal features for each token, CRF++ also provides feature templates and macro that can generat4e different composite features. There are two types of templates: Unigram and Bigram output labels. The former makes use of current output label with primal features coded by macro to generate feature functions, while the later uses previous output labels, resulting more feature functions. In this task, both types are applied.

(2) IO Tagging vs. BIEO Tagging

There are two tagging methods for information extraction. IO tagging has only two labels: "I" stands for inside the extraction target and "O" stands for outside the extraction target. To distinguish the beginning and ending tokens from tokens inside the extraction target, BIEO tagging has two additional labels for beginning and ending labels. For example, the corresponding labels for each token of the string "麒麟學堂地址:510 彰化縣員林鎮山腳路三段 2 巷 6 號" are shown in Figure 4.

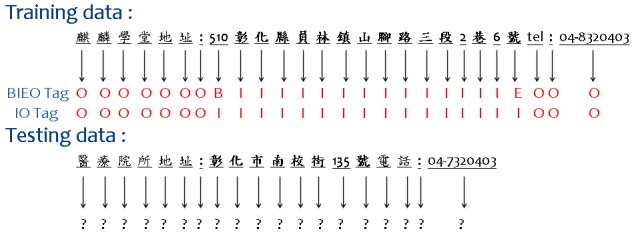

Figure 4: symbol corresponding to feature

3.5 Full Address Extraction

For BIEO tagging, we extract a segment which begins with B label followed by at least one I label and ends with an E label. For once a testing sequence has each token labeled with proper labels, the corresponding segment can be extracted accordingly.

While BIEO tagging has B and O tags which can act as sentinels for boundary detection, IO tagging lacks of such labels. Thus, it is possible that mislabeled tokens would result in small address segments which are not complete. Therefore, we try to apply maximal scoring subsequence (MSS) to the labeled sequence for full address extraction. An MSS is a contiguous subsequence having the greatest total score than all proper subsequences. The problem of MSS arises in biological sequence analysis, where the high-scoring subsequences correspond to regions of unusual composition in a nucleic acid or protein sequence. It is also applied in article text extraction from WWW in [11].

Since MSS algorithm requires the input to be a sequence of positive and negative values, we subtract the probability of each token to be I label by 0.5 to make the value to fall into an interval between [-0.5, 0.5]. For example, given a sequence of probabilities (0, 0, 0, .99, .99, .99, .40, .99, .99, .99, 0) where each value denotes the probability that a token has I label, the transformed number sequence will be (–.5, –.5, –.5, .49, .49, .49, –.10, .49, .49, .49, –.5). The task of maximal score subsequence is to find the maximal scoring subsequence (.49, .49, .49, –.10, .49, .49, .49).

Assuming a candidate segment might contain only one address, we can apply the maximum scoring subsequence algorithm in Figure 5 to extract complete address. As shown in line 8 and 9 of Figure 5, a subsequence would be extended if its sum is greater than existing maxSS. A new subsequence is enumerated if the sum of the current subsequence is less than zero (line 11-14). Such a global consideration would compensate the case of incomplete address due to mislabeled tokens. If a candidate segment might contain more than one address, we can apply Ruzzo and Tompa [12]'s algorithm to output all non-overlapping maximal scoring subsequences. Due to space limitation, the detailed algorithm is not shown here.

Algorithm 1 Maximum Scoring Subsequence

```
1: Inputs:
     S = (s_1, s_2, ..., s_n)
2: Outputs:
     maxSS
3: start = 1
4: sum = 0
5: maxSS = Ø
6: for i = 1 to n do
7:    sum = sum + s_i
8:    if sum ≥ value(maxSS) then
9:       maxSS = (s_start, s_start+1, ..., s_i)
10:   end if
11:   if sum < 0 then
12:      start = i + 1
13:      sum = 0
14:   end if
15: end for
```

Figure 5: Maximal Scoring Subsequence Algorithm

4. Associated Information Extraction

In many extraction tasks, there is an explicit need to relate the extracted entities. For example, it is not enough to find occurrences of addresses since an address without description could not support users' search requests in many applications. In contrast with other information extraction task where the extraction target is often defined by some entities, associated information is a vague idea which is intended to support users' requests on a map or other applications.

For a web page which contains a single address, the whole page can be used as a description for this web page. Thus, the task here focuses on associated information extraction from web pages which contain multiple addresses. One observation is that many web pages with multiple postal addresses often have associated information arranged regularly forming address blocks. Since the rendering is controlled by HTML tags, the HTML tags in the web page should present some regularity.

Chang and Li et al. [6] proposed an unsupervised method for associated information extraction by detecting the regularity on HTML tags. This method works well when addresses in a page are rendered in the same layout. However, address blocks might have more than one layout for displaying postal addresses (e.g. Figure 6). Therefore, in this paper we propose the idea of segmenting a page into partitions before applying the boundary detection algorithm.

Figure 6: Two different address layout in one page (purple blocks denote detected addresses)

4.1 Web page partitioning

Since the boundary detection algorithm proposed in [6] can only work when all addresses are rendered with one layout, a page needs to be processed before applying the algorithm. Formally, given a set of addresses extracted from some page, the segmentation procedure divide addresses into different clusters to distinguish address layout by their DOM tree path similarity. DOM (document object model) is a standard representation of HTML documents. Every element, label, and text in HTML is denoted by some node in the DOM tree.

In this paper, we use two methods for address partitioning (i.e. web page partitioning). The first method is based an assumption that objects with the same layout also share the same parent node, while the second method uses agglomerative hierarchical clustering for address partitioning. In both methods, we use numbered path to avoid common paths with various parents. As shown in Table 3 and Figure 7, each path is based on the composition of node numbers (instead of node names) denoting the positions of each node at the same level. Thus, each leaf node has a unique numbered path.

Leaf	DOM Tree Path	Numbered Path
a1	/html/body/div/table/tr/td	/1/2/1/1/1/1
a2	/html/body/div/table/tr/td	/1/2/1/1/2/2
a4	/html/body/div/table/tr/td	/1/2/3/3/6/4

Table 3: Numbered path for node *a1*, *a2* and *a4* in Figure 7

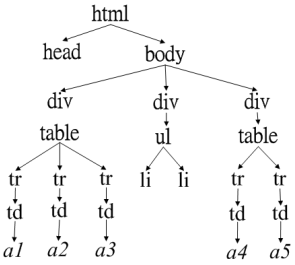

Figure 7: DOM tree example

To measure the similarity between two numbered paths, we design a formula such that higher score is given to path pairs with more common ancestors and closest neighborhood. Given one path pair with length no larger than k, nodes at level i (i<k-1) take $1/2^i$ weight of the credit, while nodes at level k-1 and k take $1/2^{k-1}$. That is, level 1 takes half the credit; level 2 takes a quarter, and so on. If the numbers at level i are equal, it will take the whole weight or a value between 0 and 1 will be assigned. Thus, the similarity between two path p and q can be calculated by

$$sim(p,q) = 1 - \sum_{i=1}^{k} \frac{1}{2^i} \frac{|p[i] - q[i]|}{\max child[i]}$$

where *p[i]* and *q[i]* denote the node number at level i and As and *maxchild[i]* denotes the number of nodes at level i. For example, the similarity for *a1* and *a3* equal to 1- (1/8 * 1/3) - (1/16 * 5/7) - (1/32 * 3/5), where 1/3, 5/7 and 3/5 are due to the closeness between two , <tr>, and <td> nodes at level 3, 4, and 5, respectively.

5. Experimental Results

The experiment contains two parts. The first part focuses on address extraction, while the second part verifies the performance of associated information extraction with the help of web page partition.

5.1 Data collection

The data set used here is prepared by querying Google with 19 administrative district names. These include Taipei (台北), Taichung (台中), Hsinchu (新竹), Taoyuan (桃園), Kaohsiung (高雄), etc. A total of 1689 web pages, including 12124 addresses are obtained. We divide the data into training set and testing set. For training set, there are 1140 web pages with 8228 addresses. For testing set, there are 549 web pages with 3896 addresses. For evaluation, we use precision, recall and F-

measures in terms of individual labels. For complete address, we calculate the values for each page and average them.

$$P_B = \frac{\#\text{ of correct B labels identified}}{\#\text{ of B labels identified}}$$

$$R_B = \frac{\#\text{ of correct B labels identified}}{\#\text{ of correct B labels}}$$

$$F = \frac{2P * R}{P + B}$$

5.2 Performance of address extraction

Before we start with the sequence-labeling task, we first determine the parameter, i.e. the window size for candidate string segmentation. Using a small window size for address suffixes would produce a lot of incomplete addresses, while a large window size would generate long candidate strings, which might result in long labeling time. Therefore, we examine the number of candidate strings generated with various number of window size using training set. We also divide the candidate strings into three groups based on the number of addresses contained in the candidate strings: the first group includes candidate strings which contains no address, the second group includes candidate strings which contain exactly one address, the last group includes candidate strings which contain more than one address.

As shown in Figure 8, the number of candidate strings drops from 23788 to 12841 when the window size increases from 2 to 8 for ICCS tokenization. More obviously, the number of candidate strings that contain more than one address increases from 23 to 426 with the window size increase from 2 to 8. A log scale for number of candidate strings is used since the numbers vary from tens to ten thousands. Note that the number of overall candidate strings seems to be large for window size 2, but many of them are incomplete address fragment. The actual number of addresses captured by these address suffixes is 7516 with window size 2. The number increases to 8228 with window size 6 and remains at the level for larger window size. Therefore, we choose a window size of 6 for ICCS.

A similar trend is observed for YCWS tokenization as shown in Figure 9. The number of overall candidate strings drops from 19889 to 11755 when the window size increases from 2 to 6, while the number of candidate strings that contain more than one address increases from 46 to 470. The actual number of addresses captured by these address suffixes increases from 8003 with window size 2 to 8228 with window size 6 and remains at the level for larger window size. Therefore, we choose a window size of 4 for YCWS.

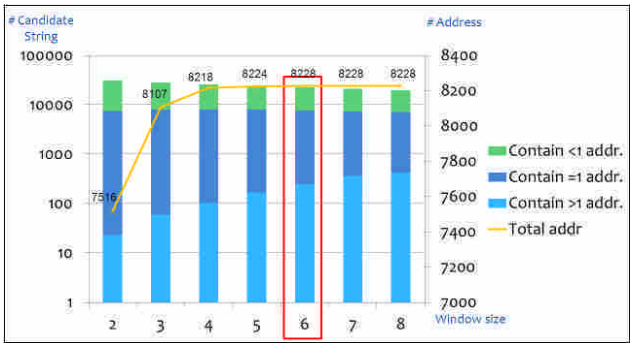

Figure 8: Number of candidate strings and addresses with various window sizes for ICCS tokenization.

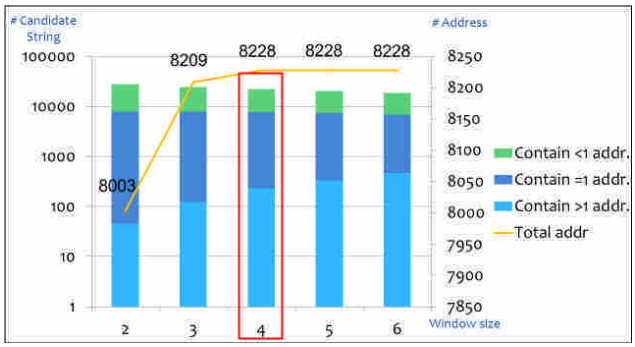

Figure 9: Number of candidate strings and addresses with various window sizes for YCWS tokenization.

BIEO Tagging	B	I	E	O
ICCS	0.9923	0.9934	0.9937	0.9983
YCWS	0.9666	0.9797	0.9780	0.9967

Table 4: F-measure of individual labeling using BIEO tagging.

IO Tagging	I	O
ICCS	0.9771	0.9940
YCWS	0.9793	0.9939

Table 5: F-measure of individual labeling using IO tagging.

Tagging	Tokenization	Avg P	Avg R	Avg F
BIEO	ICCS	0.9713	0.9689	0.9701
BIEO	YCWS	0.9689	0.9580	0.9634
IO	ICCS	0.9527	0.9464	0.9497
IO	YCWS	0.9492	0.9316	0.9403
Baseline: Regular Expression		0.8805	0.9020	0.8911

Table 6: Performance of complete address extraction.

Next, we conduct CRF learning module with two tagging methods: BIEO and IO tagging. We then test the learned models by examining their F-measure in terms of individual labeling. Table 4 and Table 5 show the experiment result with two tokenization methods for BIEO and IO tagging respectively. For BIEO tagging, ICCS has better performance than YCWS in address related tags (B, I, E) and equivalent F-measure in O tags. As for IO tagging, the performance between ICCS and YCWS are undistinguishable. However, the F-measure for address related tag (I) is less than that of BIEO. Overall, Chinese word segmentation does not provide better performance and might hurt the performance due to incorrect word segmentation.

Next, we show the performance of complete address extraction for the four combinations between two tagging methods and two tokenization methods. We have also manually constructed regular expressions for Chinese address extraction as baseline. Since ICCS tokenization with BIEO tagging has very high F-measure (greater than 0.99) on all address-related tags (B, I, E), the performance on complete address extraction is also the best (0.97 F-measure) as shown in Table 6. For the two combinations with IO tagging, the F-measure of the complete address extraction is no greater than 0.95 even with the help of maximal scoring subsequence. Some of the errors are caused by incorrect labeling of I tag between two addresses. Overall, BIEO is better

than IO and ICCS is better than YCWS since some erroneous labeling happen on incorrect segmentation.

In order to compare performance between different methods, we use t-test to examine if the average F-measure has significant differences. We individually test 4 pairs between IO+YCWS and Regular Expression, BIEO+ICCS and IO+ICCS, BIEO+ICCS and BIEO+YWCS, finally IO+ICCS and IO+YCWS. As shown in Table 7, the P value is 3.10e-8, 5.73e-6, 9.64e-3, 3.46e-2, respectively. The first two P value are smaller than 0.0001, showing that the difference are significant at confidence level 99.99%. Thus, BIEO is better than IO. Furthermore, the difference between ICCS and YCWS is also significant at confidence level of 99% for BIEO tagging (P value < 0.01) and 95% for IO tagging (P value < 0.05).

Pair of two methods		P value
IO+YCWS	Regular expression	3.10e-8
BIEO+ICCS	IO+ICCS	5.73e-6
BIEO+ICCS	BIEO+YWCS	9.64e-3
IO+ICCS	IO+YCWS	3.46e-2

Table 7: P values of t-test with various pairs.

5.3 Performance of associated information extraction

In this experiment, we test the performance of associated information extraction with the help of web page partition. The experiment is conducted with 549 web pages that contain multiple addresses in one page. A total of 3896 addresses are contained in these web pages. Table 8 shows the number of correct and incorrect boundaries identified by the corresponding methods. The accuracy is improved from 0.9040 to 0.9212 with the help of web page segmentation.

Method	Correct	Incorrect	Accuracy
Chang and Li [4]	3522	374	0.9040
With page partitioning	3598	307	0.9212

Table 8: Performance of associated information extraction.

6. Conclusions

Address information is an indispensable element in our life. An address database with associated information would better identify the entities to be discussed and highly benefit location-based services and other applications. In this paper, we use a machine-learning based approach for Chinese address extraction. We compare two tokenization methods (one based on individual character and the other based on Chinese word segmentation) and two tagging methods (one based BIEO and the other based on IO). The result shows that BIEO tagging with ICCS tokenization achieves best F-measure at 0.9701. The second part of this paper focus on associated information extraction. With the help of web page partitioning based on extracted addresses in the previous stage, more accurate associated information could be extracted. The accuracy for boundary detection is improved from 0.90 to 0.92.

References

[1] CRF++: Yet Another CRFtoolkit: http://crfpp.sourceforge.net/

[2] 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.

[3] 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.

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

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

[6] C.-H. Chang and S.-Y. Li: MapMarker: Extraction of Postal Addresses and Associated Information for General Web Pages. Web Intelligence 2010:105-111.

[7] J. Lafferty, A. McCallum, and F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. ICML 2001: 282-289.

[8] C. Lin, Q. Zhang, X. Meng, and W. Liu: Postal Address Detection from Web Documents. WIRI 2005: 40-45.

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

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

[11] J. Pasternack and D. Roth. Extracting Article Text from The Web With Maximum Subsequence Segmentation, WWW 2009: 971-980.

[12] W. L. Ruzzo and M. Tompa. A Linear Time Algorithm for Finding all Maximal Scoring Subsequences. 7th Intl. Conf. Intelligent Systems for Molecular Biology, Aug. 1999: 234-241.

[13] 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.

[14] 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.

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