Data Pre-Processing for Real-World E-Commerce Delivery Address Clustering

Yuan Zhang[1, 2, a]

[1] Shanghai International Studies University, Xianda College of Economics & Humanities, Shanghai, China

[2] Chinese Academy of Social Sciences, Shanghai Academy, Shanghai, China

[a]1601010@xdsisu.edu.cn

Keywords: E-Commerce, Data Pre-processing, Clustering, Logical Hierarchical Scraping

Abstract. Rapid growth of economy and popularization of electronic commerce have facilitated the development of logistics industry. Aiming to increase e-commerce logistics efficiency, extracting meaningful information from the complex raw data is essential. In real-world business operation, the order data should be pre-processed for the convenience of customer analysis and delivery route planning. This paper focuses on a real-world e-commerce company case, and provides an approach for pre-processing of raw address data with messy text structures.

1 Introduction

In the Internet era, electronic commerce companies obtain plenty of on-line orders. Be accompanied with large amounts of address data, the companies tend to extract and pre-process the raw data before cluster millions of address data in groups, so that the order delivering can be orderly and efficient. This paper focuses on raw address data with messy text structures in a real-world e-commerce company case, and provides an effective data pre-processing flow.

2 Data Pre-Processing Approach in Real-World Application

In this real-world case, the raw order data provided by the e-commerce organization contained detailed sales and distribution data of each order during seven months. It was necessary to extract the meaningful address data and other corresponding data from approximately 2.7 million rows of raw data. About 171 thousand non-redundant and non-null rows of data were extracted from the raw business database, including the columns of Order ID, Order Date, Delivery Store ID, Delivery Time, Shipping Address, Order Quantity and Total Amount. The Shipping Address data was combined by three columns of raw data, including address 1 (SHIP_ADDR_1), address 2 (SHIP_ADDR_2) and address 3 (SHIP_ADDR_3).

In this case, delivery address data in natural language had been converted in to valid address data which can be geocoded through Google API.

The most complicated and essential order information was order address data. Most raw address data could not be recognized by Google API because they were not in a standardized format and a unique language format. Some raw address data contained contact phone numbers, or non-address texts (such as supplementary information of delivery requirements, etc.) contained in parentheses or between asterisks, etc.

Aiming to transform the “dirty data” [5] to be a recognized format, the raw address data should be compiled by utilizing regular expression method. Regular expression is an efficient method adopted in programming for pattern matching. Regular expressions commonly provide a flexible and concise pattern to match strings of text, which are usually used for syntax highlighting systems, data validation and search engines, in order to determine an algorithmic match and search pattern to the query a user is asking.

Basing on the raw address data of e-commerce orders, the flow chart of compiling and converting the address data could be illustrated as below:

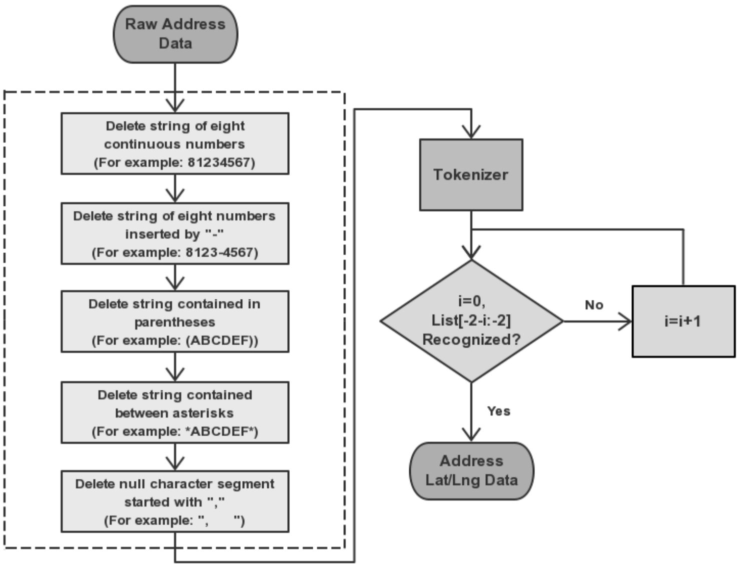
Raw Address Data
Delete string of eight continuous numbers (For example: 81234567)
Delete string of eight numbers inserted by "-" (For example: 8123-4567)
Delete string contained in parentheses (For example: (ABCDEF))
Delete string contained between asterisks (For example: *ABCDEF*)
Delete null character segment started with "," (For example: ",   ")
Tokenizer
i=0, List[-2-i:-2] Recognized?
No
i=i+1
Yes
Address Lat/Lng Data

Fig.1. The Flow Chart of Compiling and Converting the Address Data

The compiling process involved five regular expression patterns in a sequential order. The primary four steps included deleting strings of eight continuous numbers, deleting strings of eight numbers inserted by the symbol of “-”, deleting strings contained in parentheses “(” and “)” and deleting strings contained between asterisks “*”. Subsequently, as some texts were deleted in the previous four regular expression steps, the existing null character segments started with “,” should be deleted.

Commonly, a data compiling and cleaning process involves tokenization. Tokenization is the process of breaking a stream of text up into words, phrases, symbols, or other meaningful elements called tokens [1]. In the real-world case, each stream of address data was broken into several tokens by comma symbol already. Therefore, the tokenization step could be omitted in this case. After this step, the address data could be regarded as “semi-cleaned” data.

Then, the “semi-cleaned” address data would be converted into latitude and longitude data by using Google Map API. In this process, Google Map Geocoding API was used to find the LAT/LNG data of address data in text. Geocoding API supported the process of converting human-readable addresses (for example: "16B C****** Court, W****** Gardens, 2-4 ****** Street") (the exampled address is partially displayed due to confidentiality) into geographic coordinates (for example: latitude: 22.3641 and longitude: 114.1795), which could be used to place markers on a map and further study on distribution scheme.

In this conversion process, a loop was used in programming. As the last token in an address stream was not pivotal to the recognition by Google Server, the loop of recognition process could be illustrated as below:

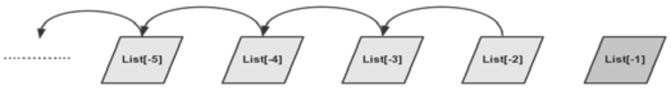

Fig.2. The Loop of Recognition Process

If the List[-2] could not be recognized successfully, List[-3:-2] would be checked; if still not recognized, List[-4:-2] would be checked through Google Map Geocoding API; and by this analogy, the loop would end until the address could be recognized successfully or the entire lists of token had been checked. Finally, the human-readable address data could be converted into geographic coordinates.

However, the free Google Map API had two limitations: 2500 requests per 24 hour period and 5 requests per second. To facilitate the conversion process, method of logical hierarchical scraping had been adopted. The processes could be structured as below:

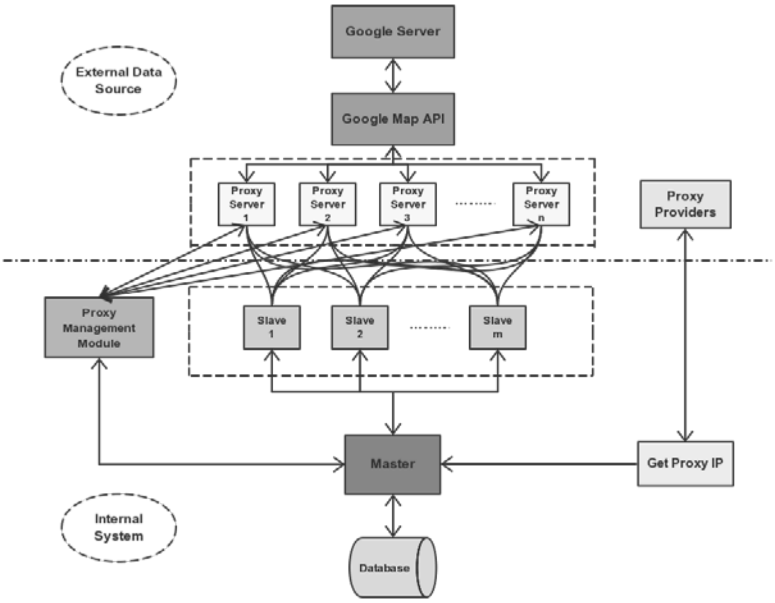

Fig.3. The Flow Chart of Logical Hierarchical Scraping

The structure could be divided into two sections: external data source and internal system. The external environment included Google Server, Google Map API, Proxy Providers and Proxy Servers; while the internal environment included Slaves, Master, Proxy Management Module, Program aiming to get Proxy IP and Database of online shopping orders.

By using Get Proxy IP module in program, Proxy IPs were extracted from several Proxy Providers. Subsequently, the Proxy Servers would be accessed by slaves. A proxy management module was established, in order to maintain the program running efficiently and effectively. Whenever, an error or a timeout occurred when the program was running, the proxy management module would order the slave to skip the former Proxy IP and try the next one.

With the consideration of the database size, the slaves were operated manually, instead of not being controlled by a master in this case. In this case, approximately 15 slaves were used to run the program and the logical hierarchical scraping program was implemented on 15 PCs which had been configured Enthought Canopy. Then the outputs could be obtained and collected.

According to the 7-months data of the company in case, the total quantity of order during the period is 170 thousands approximately. In the converted result, eventually about 142 thousand rows of order data were recognized accurately. Therefore, in this phase, the conversion accuracy of address data reached at around 85%.

It was found that among 142 thousand rows of order data, order quantity of some orders were less than 1 or equal to 0, as well as total order amount. In consideration of some negative total order goods quantity stood for reverse logistics of sales return, the orders of which the order quantity were less than 1 had been omitted in this case. After the initial data cleaning, there were 140 thousand rows of effective data left.

3 Data Manipulation and Analysis

After address data pre-processing, clustering method is adopted in this real-world case.

One technique for analyzing multi-dimensional numeric data is to attempt to group the data into clusters by using k-means algorithm. K-means [2] is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. This algorithm aims at minimizing an objective function, in this case a squared error function. The objective function

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \| x_i^{(j)} - C_j \|^2 \tag{1}$$

, where $\| x_i^{(j)} - C_j \|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster center C_j, is an indicator of the distance of the n data points from their respective cluster centers.

K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. Two good rules of thumb for the number of clusters to expect are

$$k=\sqrt{n} \text{ or } k=\sqrt{n/2} \tag{2}$$

Choosing the value of k is “more than an art than a science” [6], although there is limitation of k: k is an integer and 1≤k≤n, where n is quantity of data points, n is equal to approximately 140 thousand in this case. It was believed that more clusters would enhance the accuracy of final results. However, excessive cluster quantity would decrease the efficiency of further distribution planning process. Aiming to pursue a more accurate and authentic quantitative research result, $\sqrt{n}$ is adopted as our value of k in the K-means clustering process.

Finally, by using the tool of SPSS, addresses could be grouped into 375 clusters in this case.

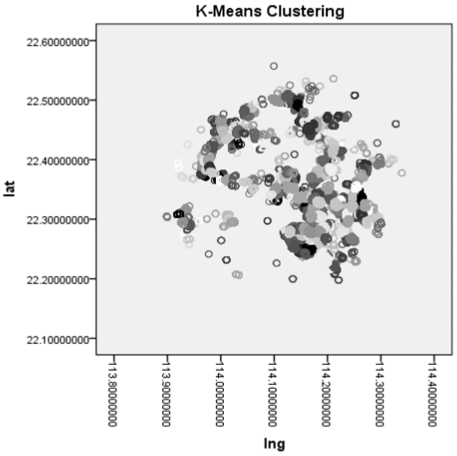

Fig.4. Visualization of K-means Clusters

4 Conclusions

In this paper, an effective data pre-processing flow has been structured. In real-world business operation, pre-processing of big data has become increasingly pivotal to modern companies. In the future study, how to further increase the efficiency of data extraction and logical hierarchical scraping process will be discussed.

Acknowledgement

This work is sponsored by “Chenguang Program” supported by Shanghai Education Development Foundation and Shanghai Municipal Education Commission (16CGB07).

References

[1] Huang, C., Simon, P., Hsieh, S., & Prevot, L. (2007) Rethinking Chinese Word Segmentation: Tokenization, Character Classification, or Word break Identification "The Art of Tokenization", Developer Works, Jan 23 (2013)

[2] J. B. MacQueen: "Some Methods for classification and Analysis of Multivariate Observations, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability", Berkeley, University of California Press, 1:281-297 (1967)

[3] Mattos Ribeiro G, Laporte G. An adaptive large neighborhood search heuristic for the

cumulative capacitated vehicle routing problem [J]. Computers & Operations Research, March, Vol 39(3) (2012), p. 728-735

[4] Maurice C, Kennedy J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space [J].IEEE Transactions on Evolutionary Computation, February, Vol 6(1) (2012), p. 58-73

[5] Zhengping Li, Ruisheng Wang, Hongwei Liu, Wenfeng Zhou, Xiangsun Zhang. Models and Algorithms for the Constrained Orienteering Problem [J]. Lecture Notes in Operations Research: Operations Research and Its Applications Vol. 12 (2010), p. 52-60

[6] Zhengping Li, Qingyun Xu, Na Li, Yuanguan Ma. Mathematical Model and Solving Method Based on Software for the shortest Time Limited Transportation Problem [C]. Lecture Notes in Operations Research: Operations Research: Operation Research and its Applications, Vol. 12 (2010), p. 389-396