Application Research and Simulation Analysis of Power Address Mining Based on Address Fuzzy Matching Algorithm

Jian Guo *

State Grid Yancheng Power Supply Company, Yancheng, 224005, China

Abstract: Address recognition, as one of the important scenarios of natural language processing in big data applications, is an important and extremely practical technical means. Currently, the evolution of big data applications is being actively promoted, and more and more people are using big data technology to empower power address recognition. Address information in many data assets is the core area of connected devices, and the analysis and mining of core algorithms has extremely high value. This paper first analyzes the application scenarios involved in the common address information of electric power, and formulates the extraction and matching method according to the key points of the core application scenarios and the address recognition requirements; then, based on the data samples, the accuracy and calculation speed of the address recognition method are studied, and the Algorithms are analyzed and compared. Practicality; finally summarize the algorithm, and look forward to ways to improve the algorithm in the future.

Keywords: Natural language processing; Fuzzy matching; Address recognition; Power; Big data

1. Introduction

Various research evidences show that human activity information is basically related to geographic location. The same is true in all businesses of electric power companies, where electricity use addresses, repair work orders, equipment locations, etc., are the core fields of geographic location. At present, the conventional mode is basically to manually extract the address fields contained in various types of information for matching, which is a relatively cumbersome and time-consuming task for the relevant staff of electric power enterprises. If the geographic location information contained in relevant internal business and customer demands can be quickly and accurately identified, it can well assist power companies to quickly respond to the needs of all parties, improve service capabilities, and enhance the value of data assets. The following aims to explore and practice how to automatically, quickly and accurately perform geographic location matching. Based on the geographic location of various common information, realize the automatic extraction and matching of geographic location, and write a variety of address extraction and matching methods based on Python, and comprehensively consider the accuracy of matching. The two types of indicators, namely, power rate and operating speed, explore methods that are more suitable for power companies.

2. Research background

2.1 Technical overview

The earliest research work on natural language understanding was machine translation. The design of machine translation was first proposed by American Weaver in 1949. In the 1960s, there were large-scale research work on machine translation in foreign countries, which cost a huge amount of money; but at that time, people obviously underestimated the complexity of natural language, and the theory and technology of language processing were not popular, and the overall progress of machine translation slow. At that time, the main method was to store the words and phrases of the two languages to create a large dictionary corresponding to one-to-one correspondence during translation. Technically, it was only to adjust the same order of the language, but the performance was poor when faced with daily semantic communication.

Under normal circumstances, most of them can be resolved according to the corresponding context and scenario. In other words, for individuals, it does not cause incomprehensible situations. This is a common contextual analysis. But in order to understand the semantics correctly, it is necessary to reason based on a huge amount of basic common sense. How to collect and sort these basic common sense in a relatively complete manner, how to store it in the computer quickly and well in a suitable

[*] Corresponding author: JianGuo224005 @whu.edu.cn

way, and how to reasonably use them to parse semantics is a huge workload and Very difficult job. This is not a task that can be completed in a short period of time. Long-term and systematic exploration and research are still needed. At present, natural language processing applications include "language segmentation", "semantic analysis", "language translation", etc. [2], and fuzzy recognition is one of the important method ideas in this field, which refers to the identification of large amounts of complex information The useful part is to recognize the correlation between the received information and past memories and experiences, and eliminate irrelevant information. Regardless of the object (language, text, image, etc.), the core points that need to be recognized have their own unique key features [3]. For Chinese addresses, its core feature is the keywords in the address, such as " The common method is to effectively locate the address in the article based on keywords, and simulate and match the address based on word segmentation, comparison, verification, etc., to achieve the extraction of article address informa

2.2 Current status of power companies and address-related businesses

Address-related businesses of power companies include customer new installations, repairs, and equipment installation management. In addition to the current inaccuracy and irregularity of geographic location information, there are also problems such as long time-consuming and heavy workload for address extraction and matching. From the perspective of scenario applications, it can be roughly divided into the following three types of scenarios:

1) Entry of work conditions such as internal maintenance, inspections, and power outages of power companies. For example, when professional operation and maintenance of equipment such as power transmission and transformation are carried out, the working conditions of the day need to be entered into the system, especially when a fault is found, the fault content (including the cause of the fault, the location of the fault, etc.) needs to be entered in the system. The content is mainly filled in manually .

2) Handling of external customer appeals of power companies. For example, the 95598 order is based on the customer's request to form the detailed content of the work order. Before the work order is dispatched, the standard format of the work order needs to be drafted. As shown in Table 1, it is necessary to manually complete the content of the jurisdiction and address information.

Table 1 Basic information

Power supply unit	Affiliated district/county	business type	Business subtype	Processing status	Contact address	Acceptable content
Chengbei Customer Service Center	Yancheng	Fault report	Resident service internal failure	Archive	128 Guoan Avenue, Yancheng City, Jiangsu Province	1. GPRS module failure

3) Big data application, various professional management departments carry out big data analysis according to professional management needs, and many scenarios need to carry out related work in conjunction with address information. For example, the operation inspection needs to formulate the next step of the distribution network reliability improvement plan based on the location of the equipment failure; the marketing department needs to carry out targeted marketing services based on the content of the complaint to improve the quality of marketing services. If there is no accurate address information, it is necessary to sort out the geographic location by manual observation and positioning based on fault content information or irregular address information. If the workload is large, the granularity of data analysis may be discarded to a certain extent.

The first two application scenarios are the work content that must be involved in the workflow. Although it seems simple, a large amount of solid work also inadvertently consumes huge human resources, and the third situation will restrict power companies. The development of lean management. In addition to the above three types of scenarios, with the continuous evolution of the big data process of power companies, data mining for address information is also increasing. Therefore, there is an urgent need for efficient and accurate address recognition methods.

3. Data preparation

3.1 Article information

Prepare Chinese address information generally as "province-city-county-district-street-street number", and the content of articles that are often contacted daily mainly include the following types of samples

1) Type of web page information filling. Its performance is that the format is relatively standardized, there is basically no multiple address information, and the location markers for identifying the address information are obvious, as shown in Table 2.

Table 2 Example of webpage information filling

Basic Information		
Service object	Natural person	Office type
Statutory closing time limit	30	Statutory deadline unit
Promise to close the time limit	5	Promise to close the time limit unit
Approval results	License	Approval result name
Whether to charge	no	Lawyer's License
General scope	without	Processing form
Handling place	6 Liuxiao Road, Yancheng City	Special link (special procedure)

2) Information disclosure type. Generally, they are notices and announcements. The format content is more formal, and the address information is clear, but multiple addresses may appear [5]. An example of information

disclosure is shown in Figure 3. 3) Article information category. Generally, news reports, customer requests, etc., with a large amount of text, irregular address information, and multiple address information, etc., are relatively complicated. For example, "Today, planned maintenance and power outages will be carried out in Wuhou District of Chengdu. The power outages will cover Changshou Road, Tongzilin Community, Chengdu Traditional Chinese Medicine Hospital, Sichuan Provincial Gymnasium... Chengdu Evening News will release a message at 12:00." After pre-collection and sorting, more than 500 pieces of article address information were collected, and the 3 common situations were integrated to make the matching samples as close to the real application scenario as possible to ensure that the algorithm is operability and practical in practical applications.

3.2 Address library information preparation

Through online and offline data collection, 630,000 address databases were collected to ensure complete coverage of the article address information that needs to be matched in the sample range of the address database, and similar and similar addresses were added to further approximate the actual application situation. The address library data is shown in Table 3

Table 3 Address library data display

province	city	street	street_ num	locationx	locationy
Jiangsu Province	Yancheng	Jieguang Road	No. 561	115.3678097	33.27176266
Jiangsu Province	Yancheng	Linquan Road	No. 428	1158003975	32.8978435
Jiangsu Province	Yancheng	Paopu Street	51	115.811045	32.90747175
Jiangsu Province	Yancheng	Yingzhou Middle Road	No. 98	115.8251926	32.894331 6
Jiangsu Province	Yancheng	Qinghe East Road	571-3	115.8110317	32.89035383
Jiangsu Province	Yancheng	Guangming North Road	No. 199	115.263916	33.06525
Jiangsu Province	Yancheng	GuheEast Road	No.42	1155918	32.638973

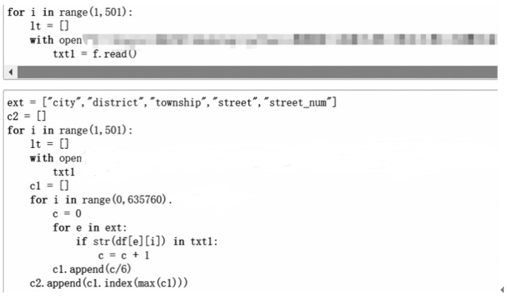
Remarks: The running configuration of this algorithm test is 2.6GHz/CPU, and the memory is 16GB. According to the requirements of data confidentiality, the address database data is only shown in screenshots, and the coordinates shown here have been adjusted. They are not real coordinates, but they will not affect the research content.

4. Algorithm ideas and implementation

4.1 Scheme 1: Address information search weighting

It has been verified to directly use regular expressions to formulate the extraction style [6], it is difficult to extract the address, especially the street number, so the idea is as follows: the content of the article is directly included in the address database for matching, corresponding to the 6 fields in each row of the address database, The matched field is 1, and the unmatched field is 0. After adding up, dividing by 6 is the matching rate of the address. Finally, the address with the highest matching rate is found as the matching address of the article. The basic code is implemented as follows: Data read: read the address library data into the Jupyter runtime environment.

Search weighted one by one: read the information of 500 articles in a loop, and incorporate the address library field into the article one by one for comparison. If the field appears in the article, the address will be incremented by 1, and the address information with the largest value will be selected as the matching result , The code is shown in Figure 1.

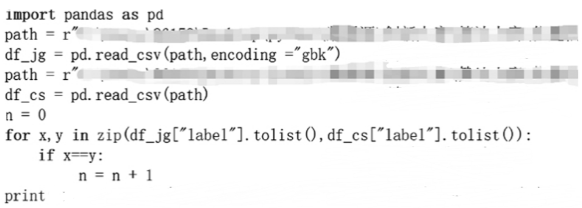


```
for i in range(1,501):
    lt = []
    with open[redacted]
        txt1 = f.read()
```

```
ext = ["city","district","township","street","street_num"]
c2 = []
for i in range(1,501):
    lt = []
    with open
        txt1
    c1 = []
    for i in range(0,635760).
        c = 0
        for e in ext:
            if str(df[e][i]) in txt1:
                c = c + 1
        c1.append(c/6)
    c2.append(c1.index(max(c1)))
```

Figure 1 Address search weighted realization

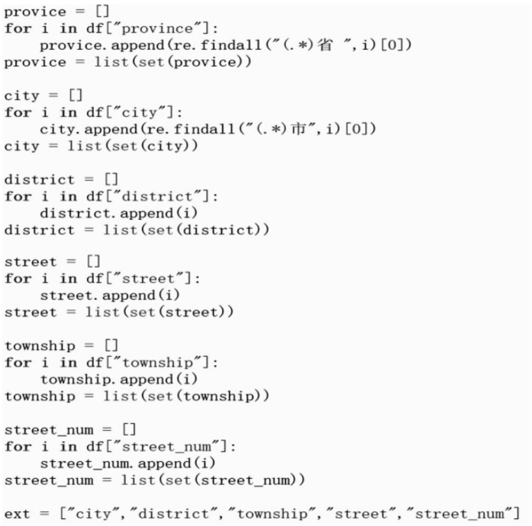
Result comparison: cyclically extract the test results and compare and verify the correct answer table, calculate the number of correct results and the accuracy of the algorithm, the code implementation and running results are shown in Figure 2.



```
import pandas as pd
path = r"[redacted]
df_jg = pd.read_csv(path,encoding ="gbk")
path = r"[redacted]
df_cs = pd.read_csv(path)
n = 0
for x,y in zip(df_jg["label"].tolist(),df_cs["label"].tolist()):
    if x==y:
        n = n + 1
print
```

Figure 2 Scheme 1 verification of calculation results

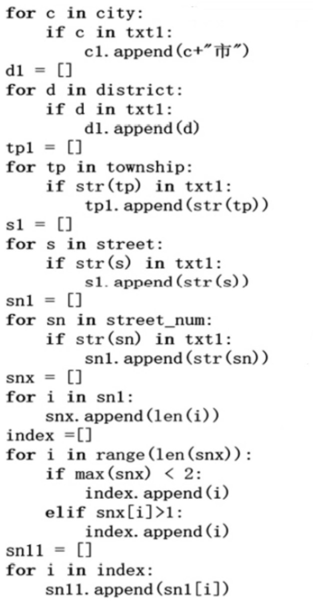
From the calculation results, the accuracy of this method is 71.21%, but from the calculation speed, it takes 4 hours to complete all matching for 500 articles, and the average time is about 0.5 minutes for each article, which is completely unable to reach the practical level. 3.2 Scheme 2: Simulating manual address screening After the experiment of the previous method, although the accuracy can basically reach the optimized conditions, the operating efficiency is low, so it is necessary to re-adjust the thinking. After analysis, the main reason for the long time-consuming option 1 is that each article needs to be matched with 630,000 pieces of data in the address database, which requires a large amount of calculations. In fact, there is no need to match 630,000 entries each time. On the one hand, the fields can be de-duplicated. For example, there are only 6 provinces in the address database, 80 cities, and 1,000 counties. Therefore, it is possible to continuously locate provinces-cities-counties-Area, gradually narrow the matching range, so as to greatly reduce the complexity of the calculation. From the above ideas, this method is basically similar to manual screening (extract key field information one by one, and then filter in the address database, continuously narrowing the scope to achieve the final positioning). The basic implementation is as follows: 1) Based on the deduplication function of the Python collection, the province, city, county, district, street, street number and other fields of the address library are merged and deduplicated to form a deduplication list for each field of the address library. , The code is shown in Figure 4.



```
provice = []
for i in df["province"]:
    provice.append(re.findall("(.*)省 ",i)[0])
provice = list(set(provice))

city = []
for i in df["city"]:
    city.append(re.findall("(.*)市",i)[0])
city = list(set(city))

district = []
for i in df["district"]:
    district.append(i)
district = list(set(district))

street = []
for i in df["street"]:
    street.append(i)
street = list(set(street))

township = []
for i in df["township"]:
    township.append(i)
township = list(set(township))

street_num = []
for i in df["street_num"]:
    street_num.append(i)
street_num = list(set(street_num))

ext = ["city","district","township","street","street_num"]
```

Figure 3 Core address field de-duplication

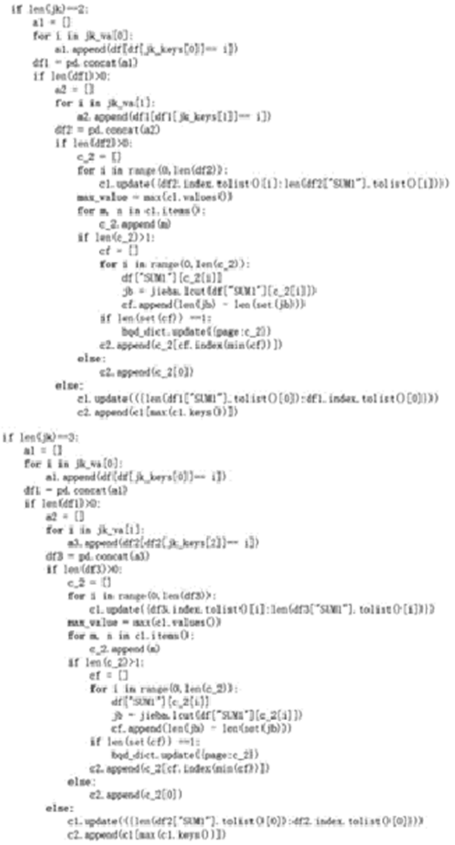
2) Put the address library address information list into the address library for cyclic matching. If the matching is successful, it will be extracted to form a matching result dictionary. The code is shown in Figure 5.



```
for c in city:
    if c in txt1:
        c1.append(c+"市")
d1 = []
for d in district:
    if d in txt1:
        d1.append(d)
tp1 = []
for tp in township:
    if str(tp) in txt1:
        tp1.append(str(tp))
s1 = []
for s in street:
    if str(s) in txt1:
        s1.append(str(s))
sn1 = []
for sn in street_num:
    if str(sn) in txt1:
        sn1.append(str(sn))
snx = []
for i in sn1:
    snx.append(len(i))
index =[]
for i in range(len(snx)):
    if max(snx) < 2:
        index.append(i)
    elif snx[i]>1:
        index.append(i)
sn11 = []
for i in index:
    sn11.append(sn1[i])
```

Figure 4 Article address information extraction

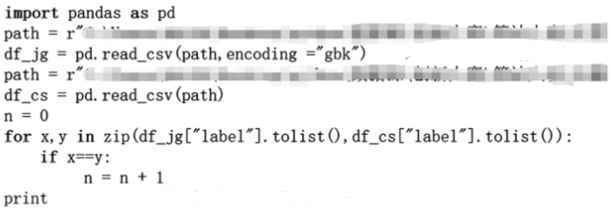
3) According to the matching result dictionary, the number of matching fields is included in the address database for positioning. For example, if "city-street-street number" is matched, the number of the matching field is 3, and the corresponding 3 fields in the address database will be screened 3 times , And gradually narrow the matching range to complete the location of the address, as shown in Figure 6.



```
if len(jk)==2:
    a1 = []
    for i in jk_va[0]:
        a1.append(df[df[jk_keys[0]]== i])
    df1 = pd.concat(a1)
    if len(df1)>0:
        a2 = []
        for i in jk_va[1]:
            a2.append(df1[df1[jk_keys[1]]== i])
        df2 = pd.concat(a2)
        if len(df2)>0:
            c_2 = []
            for i in range(0,len(df2)):
                c1.update({df2.index.tolist()[i]:len(df2["SUM1"].tolist()[i])})
            max_value = max(c1.values())
            for m, n in c1.items():
                c_2.append(m)
            if len(c_2)>1:
                cf = []
                for i in range(0,len(c_2)):
                    df["SUM1"][c_2[i]]
                    jb = jieba.lcut(df["SUM1"][c_2[i]])
                    cf.append(len(jb) - len(set(jb)))
                if len(set(cf)) ==1:
                    bqd_dict.update({page:c_2})
                c2.append(c_2[cf.index(min(cf))])
            else:
                c2.append(c_2[0])
        else:
            c1.update({(len(df1["SUM1"].tolist()[0]):df1.index.tolist()[0])})
            c2.append(c1[max(c1.keys())])

if len(jk)==3:
    a1 = []
    for i in jk_va[0]:
        a1.append(df[df[jk_keys[0]]== i])
    df1 = pd.concat(a1)
    if len(df1)>0:
        a2 = []
        for i in jk_va[1]:
            a3.append(df2[df2[jk_keys[2]]== i])
        df3 = pd.concat(a3)
        if len(df3)>0:
            c_2 = []
            for i in range(0,len(df3)):
                c1.update({df3.index.tolist()[i]:len(df3["SUM1"].tolist()[i])})
            max_value = max(c1.values())
            for m, n in c1.items():
                c_2.append(m)
            if len(c_2)>1:
                cf = []
                for i in range(0,len(c_2)):
                    df["SUM1"][c_2[i]]
                    jb = jieba.lcut(df["SUM1"][c_2[i]])
                    cf.append(len(jb) - len(set(jb)))
                if len(set(cf)) ==1:
                    bqd_dict.update({page:c_2})
                c2.append(c_2[cf.index(min(cf))])
            else:
                c2.append(c_2[0])
        else:
            c1.update({(len(df2["SUM1"].tolist()[0]):df2.index.tolist()[0])})
            c2.append(c1[max(c1.keys())])
```

Figure 5 Filtering and matching based on address extraction information

4) Verify the accuracy of the method, see Figure 7 for details.



```
import pandas as pd
path = r"[redacted]
df_jg = pd.read_csv(path, encoding ="gbk")
path = r"[redacted]
df_cs = pd.read_csv(path)
n = 0
for x,y in zip(df_jg["label"].tolist(),df_cs["label"].tolist()):
    if x==y:
        n = n + 1
print[redacted]
```

Figure 6 Verification of the calculation results of Scheme 2

The program has a significant improvement in running speed. 500 articles are matched based on 630,000 address libraries. It only takes 5 minutes to complete the operation. It takes only 0.6s to match each article on average. It fully meets the practical requirements in terms of running speed. In terms of accuracy, the program at this stage is only about 65%, which cannot reach the level of practicality (generally designed algorithms, the confidence rate should reach more than 95%). After solving the problem of slow running speed, summarize the reasons for the wrong matching, solve the wrong matching problem one by one, and optimize the algorithm. After analyzing one by one, the reasons for the error are as follows: 1) The error rate is high when the address information is extremely similar. For example, xx District-No. 16 Chengxi Street and xx District-No. 6 Chengxi Street. 2) More than two answers may be screened out. For example, the article is Chengxi No. 5, and the address database is Chengxi Street, Chengxi District, and Chengxi No. 5, Chengxi District. 3) Individual answers do not match. For example, if the article is xx street in Wuhou District, Chengdu (the street name is also in Xi'an), the result may not be output correctly due to the matching order problem (because of the problem that the city and the street cannot correspond), etc. The wrong matching result is shown in the figure 7 shown.

4.2 Scheme 3: Simulate manual address screening (optimization of scene problems)

For inaccurate matching due to problems such as address similarity and repetition, the conventional solution can be word segmentation before matching; however, due to the high complexity of address names and many unusual names [7], the commonly used cpca library cannot be used, even for word segmentation. The most effective jieba word segmentation method [8] is still difficult to accurately segment the address name. Therefore, it can only be solved by scene design. Since all the inaccurate address problems are in the "street number" field, it can be processed from the character length of the "street number". The pseudo-code implementation:

1) If the longest street number is also less than 1, then directly include the result that needs to be matched;

2) If they are not all short values, then data with a street number length of 2 or more will be included in the result that needs to be matched to avoid too much invalid data entering the content that needs to be matched;

3) Make further judgments to avoid repeated occurrences of No. 3 and No. 33. The code implementation is shown in Figure 8.

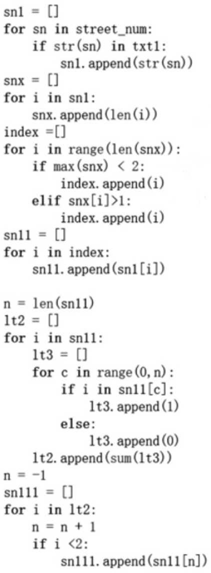
For the situation where there are multiple address information, such as Chengdu-64 and Xi'an-223, the problem is that the front and back do not correspond to each other. The solution is to locate the location where the matched street appears in the article and the location where the street number appears. Comparing the distance between the two positions, the two are similar as a group, and the accurate association of the fields before and after the address can be realized. The pseudo code is as follows:

1) Find the position strlc of the street to be matched in the article;

2) Find the nearest street number corresponding to the street;

3) Extract the street-street number for combination;

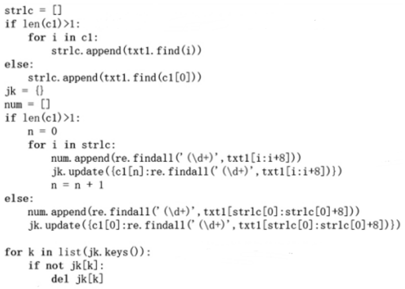
4) Incorporate the combined results into the address library for screening.



```
snl = []
for sn in street_num:
    if str(sn) in txt1:
        snl.append(str(sn))
snx = []
for i in snl:
    snx.append(len(i))
index =[]
for i in range(len(snx)):
    if max(snx) < 2:
        index.append(i)
    elif snx[i]>1:
        index.append(i)
snl1 = []
for i in index:
    snl1.append(snl[i])

n = len(snl1)
lt2 = []
for i in snl1:
    lt3 = []
    for c in range(0,n):
        if i in snl1[c]:
            lt3.append(1)
        else:
            lt3.append(0)
    lt2.append(sum(lt3))
n = -1
snl11 = []
for i in lt2:
    n = n + 1
    if i <2:
        snl11.append(snl1[n])
```

Figure 7 Preprocessing of article address extraction

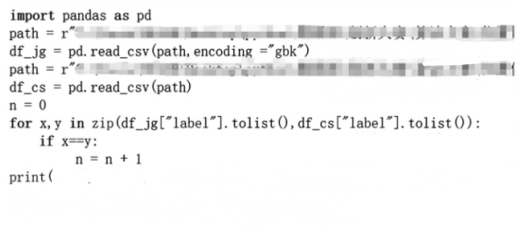
Match the street name that appears in the article, locate the position where the street name appears in the article, and find the street number closest to the street name based on that position. The code implementation is shown in Figure 10.



```
strlc = []
if len(c1)>1:
    for i in c1:
        strlc.append(txt1.find(i))
else:
    strlc.append(txt1.find(c1[0]))
jk = {}
num = []
if len(c1)>1:
    n = 0
    for i in strlc:
        num.append(re.findall(' (\d+)', txt1[i:i+8]))
        jk.update({c1[n]:re.findall(' (\d+)', txt1[i:i+8])})
        n = n + 1
else:
    num.append(re.findall(' (\d+)', txt1[strlc[0]:strlc[0]+8]))
    jk.update({c1[0]:re.findall(' (\d+)', txt1[strlc[0]:strlc[0]+8])})

for k in list(jk.keys()):
    if not jk[k]:
        del jk[k]
```

Figure 8 Locate the position of the street string in the article

Through comparison, it can be seen that after the algorithm is optimized, the number of matched streets has been greatly reduced. At the same time, a one-to-one correspondence between street names and street numbers has been realized for articles with multiple addresses, which can avoid the situation of incompatibility due to inconsistencies between the top and bottom. . The calculation result verification is shown in Figure 9. It has been verified that although the "street-street number" corresponding to Figure 9 is achieved after the optimization in Figure 9, the results of multiple streets and street numbers before optimization do not correspond and the operating speed is not accurately located, the optimized algorithm requires 10 minutes to run, which is not optimized It is twice as high as the previous one, but it also basically meets the application requirements. At the same time, the accuracy rate is increased to 98.6% after optimization, which fully meets the actual application requirements.



```
import pandas as pd
path = r"[redacted]
df_jg = pd.read_csv(path, encoding ="gbk")
path = r"[redacted]
df_cs = pd.read_csv(path)
n = 0
for x, y in zip(df_jg["label"].tolist(), df_cs["label"].tolist()):
    if x==y:
        n = n + 1
print(
```

Figure 9 Scheme 3 calculation result verification

In addition to the above three schemes, in the early stage, through consulting related documents and combining the subject content for practice, methods such as pure regular expression extraction and cpca address library extraction were eliminated. The third solution basically meets the application requirements of related business scenarios in terms of accuracy and operating speed. At the same time, the Python-based Pyinstaller code package can be directly used on various working computers of electric power enterprises, with high operability and flexibility. Table 3 shows the comprehensive comparison of the three schemes.

Table 3 Comprehensive comparison of 3 schemes

Algorithm	Accuracy/%	Running speed	Overview
Address search weight	71	4h	Slow speed and low accuracy
Simulated manual screening	65	5 min	Extremely fast and low accuracy
Simulate manual optimization	98.6	10 min	Fast speed and high accuracy

5. Conclusion

After testing, the proposed address extraction and matching method has a certain degree of practicability and can be applied to related business scenarios of electric power enterprises. The advantages of performance, speed and flexibility.

In terms of shortcomings, the solution does not design and deal with the situation that the address information that may be encountered daily is not included in the address library, so in the future, the "recent address extraction method" can be further added to the above solution. The ideas at this stage are as follows: 1) Analyze each address information of the address library based on one-hot encoding, and quantify the text; 2) divide the processed address library by districts and counties to reduce the amount of model calculations and increase the number of models ; 3) Locating the latitude and longitude weight of each field in the address library in the partition based on random forest, artificial neural network, etc.; 4) The address information of the article that fails to be directly matched is processed by one-hot processing, and is included in the model according to the region. Calculate to get its latitude and longitude; 5) According to the latitude and longitude calculated by the model, calculate the Euclidean distance or Manhattan distance from the address library to find the closest address. In the next step, we will continue to improve the comprehensiveness and usability of the proposed scheme, and provide reference for the application of management and service by electric power companies.

References

1. Chen Kaichang. Research on Chinese word segmentation in natural language processing technology[J]. Information and Computer (Theoretical Edition), 2016(19): 61-63.
2. Han Chengcheng, Li Lei, Liu Tingting, et al. Semantic text similarity calculation method[J]. Journal of East China Normal University (Natural Science Edition), 2020(5): 95-112.
3. Zhang Lei. A review of text classification and classification algorithms[J]. Computer Knowledge and Technology, 2016(34):225-226.
4. Wang Chunliu, Yang Yonghui, Deng Fei, etc. A review of text similarity calculation methods[J]. Information Science, 2019(3):158-168.
5. Xu Peizhi, Liu Xiaochun, Qin Shouke, et al. Web page information extraction method and device: CN108399167A [P]. 2018-08-14.
6. Wang Ruibo, Wang Yu, Li Jihong. Regularized cross-validation method for text data[J]. Journal of Chinese Information Processing, 2019(5): 54-65.
7. Zhang Dawei. Geographic location information extraction method of microblog data: CN105069071A [P]. 2015-11-18.
8. Liu Haoyu, Li Zhe, She Zuochao, et al. Research and practice of address fuzzy matching in the application of electric power enterprises[J]. Sichuan Electric Power Technology, 43(6):7.