PBECV: a fast resume extracting framework based on writing style recognition

Resume extraction

ABSTRACT

In the information age, companies receive thousands of resumes from job seekers everyday. Most of resumes are wrote in different format, including font size, font color and cells. As a result, it's difficult to structure these data with a general extracting method. In this paper, we propose PBECV to extract the resume information from text file without format information. Our appoach consider the writing style of each resume as the latent pattern, which help to segment resume text into different blocks and easy to parse. The experimental results on the real world data-set of resumes in Chinese show that our approach can reach the performance of algorithms that trained with the format information and the proposed approach's algorithm complexity is $O(NlogN)$. We made our demo system public for future research in the same area at http://t.cn/RvardcW

General Terms

Algorithms, Design, Experimentation

Keywords

resume information extraction, structure resume data

1. INTRODUCTION

In the information age, head-hunting companies collect millions of resumes to occupy more market share. However, most of resumes are not wrote with the standard format or follow some special template file. In order to improve the success rate of recommending some person to fit the requirements of employer, those resumes should be parsed exactly and detailed. This helps headhunters to easily and quickly search for the right candidate. The challenge is how to analysis the different kinds of resume to get the detailed information.

However, resumes are easier to structure than other texts, such as news. Different people have different writing style about personal resume, but the content of these are all around the same topic, their personal information, which contains contacts, educations, work experimences and so on. As a result of this, resumes can be segmented into servel groups, which is one of the basic ideas to solve the problem. In other words, resumes share the document-level hierarchical contextual structure [**?**].

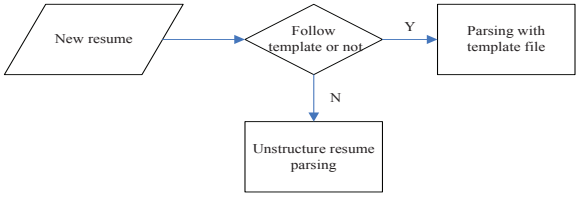

Figure 1: The cooperation of two methods

There are three main methods to deal with resumes in the practical engineering. Firstly, since many engineers has the knowledge background about how to parse a web page based on the DOM structure, they treate the resume text as a web page to extracte the details. In particular, some big recruiting platform like Monster[1] and LinkedIn[2] provide many beautiful template which make many resumes follow them. This kind of resumes can be parsed through special template file or regex rules.

Secondly, as the result of hard extracting, key word extraction approach are good to be an alterative choice. This method use search technology to query keyword from resume to check whether it match the require.

Thirdly, some researchers treat the resume extracting task as a sequence label task. The resume text can be supposed to be a mixed information heap, which contains the basic information about the person. So that this task is transfered to label the words attribute and line attribute.

In this paper, we aim to propose a rapid and effective framework to extract the detailed information from resumes. This framework can work with the methods based on template file very well to increase the accuracy of extracting task as shown in figure 1.

We consider that everyone has his/her writing style about

[1]http://www.monster.com/
[2]http://www.linkedin.com

the resume, which means that there are some latent format information during the text.

The rest of paper is organized as follows. In Section 2, we disscuss the related work. In Section 3, we explain our approach. In Section 4, experimental results are presented and analysised. Conclusion and future work are provided in Section 5.

2. RELATED WORKS

In this section, we review some of the popular methods about resume extracting. The first group of methods are based on template file. Jsoup[3] and Apache POI[4] can be used to parse resumes that follows some template file. Jsoup is a Java library for working with real-world HTML. It provides a very convenient API for extracting and manipulating data, using the best of DOM, CSS, and jquery-like methods. It also implements the HTML5 specification, and parses HTML to the same DOM as modern browsers do. The Apache POI is a useful Java library for working with Office file, based on the Open XML standards which proposed by Microsoft company. It's easy to create a specific program to extract the information from those resumes which follow the specific template file.

The second group of methods treat the resume extracting work as the nature language processing work. In [**?**], a cascaded information extraction framework was designed to support automatic resume management and routing. The first pass is used to segment the resume into cnsecutive blocks with labels indicating the information types. Then detailed information, such as Name and Address, are identified in certain blocks without searching globally in the whole resume. In [**?**], a system that aids in the shortlisting of candidates for jobs was designed. The part of parsing resume combines three technologies as follows. Table analysis is used to detect the type of values in table. CRF model is used to segment the resume text into different blocks. Content Recognizer mines the named entities salient to candidate profile.

The third group of methods treat this as key words retrieval task. In [**?**] and [**?**], only the specific data is extracted to filter the resume streams. Both of them are aim to accelerate the efficiency of search candicates for the job. Some of the important queries were created to filter the resume set, so that this can help to improve the working efficiency of the staff.

3. OUR APPROACH

In this section, we explain the details of our approach. The process can be devided into three part. First, some necessary preparations are done to the origin resume file. We converted the resume file into text format no matter what the origin format is, which is supported by Apache Tika[5]. Second, writing style is used to identify the appropriate block of each resume. Third, name entities are matched to the candicate profile based on the information statistics of the content of all the resumes in the data set.

[3]http://jsoup.org
[4]http://poi.apache.org
[5]http://tika.apache.org/

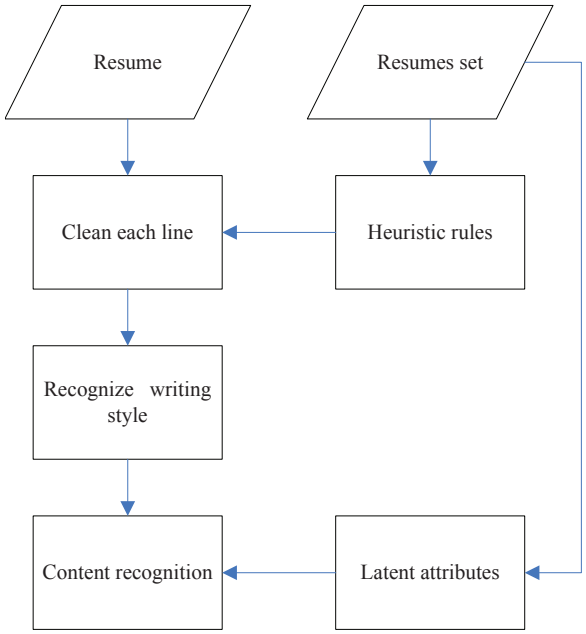

Figure 2: The framework structure

3.1 Framework Structure

Figure 2 outlines the structure of our framework. It can be devided into three parts: prepared processing, which focus on clean each line of the raw data, writing style recognition and content recognition. Before the parsing task, basic knowledge data set should be constructed from the whole set of resumes which can be seen as unsupervised text mining.

3.2 Prepared processing

From the Figure 3, it's clear to know that the raw resume text is not suitable to process directly. There are a lot of noises among the lines in each text file, such as continues blank, wrong newline, the necessary space missing. All these noised should be cleaned before the main part of extracting resume information module. We suppose the distribution of resume accordance with normal distribution, that most people will not cause serious errors on text format. Since we have millions of resume, it's easy to statistics the most common structure of sentence, especially the sentence begin with date or number. According to these rules, three kinds of operation are made, shown in Table 1.

Merge means this line should be merged with the next line.

Split means this line should be split into two lines.

Short means the blanks in this line should be remove.

We also defined three kinds of line type to facilitate the follow-up work. These three types provide the basic sentence structure which is helpful to identify the writing style.

Simple means this line is a short text and may contains few blanks.

KeyValue means this line follows the key and value struc-

Figure 3: Samples of the resume set

Table 1: heuristic rules for cleaning data

heuristic rules	operation
multiple continuous blank	short
value pair	short
begin with date pair	split
begin with part of date	merge
begin with block key words	split
begin with comma	merge
short text end with comma	merge

ture, with comma punctuation.

Complex means this line is a long text, which contains more than one punctuation.

3.3 Writing style recognition

After cleanning up the noise of raw lines, lines of resume text are devided into blocks such as basic information, education, work experiments and so on. It's not hard to find that there are some latent pattern in education and work experience block, which often has more than one item. Everyone write his/her resume will follow the local format, such as "2005-2010 [company name] [job position]", "[company name] [job position] [working time]", "[university] [major] [degree] [time range]". These local format forms the writer's personal writing style, and the writer will follow the same format during the same block, which inspaired us to identity the blocks through the writing style.

Entities are introduced into writing style recongined and in this applicantion sence simple name entity is enough, Which means for a continous text we just need know whether this is a date range entity or company name entity or university name entity. The punctuations between the continous text plays an important role in recognitize the writing style. For each line, we only detect whether this line contains date entity or some basic entity like school name, job position, company name. Each line can be transfered into entities pattern mode, as show in Figure 2. It's easy to cut the lines into blocks with the help of entities pattern and the algorithm complexty is O(n).

3.4 Attributes Match

Instead of labeling too much data, we did a lot of statistic work to collect the name entity candicates key, which often shown in the text with key value pair with the attribute name. The similarity of the entity can help to do attribute cluster, then they can be labled to the standard attribute name. The process are as follows. First, each resume is processed as the Prepared processing section 3.1 descripted. Second, those lines with key-value structure are considered to be the candicate attribute. Third, after removing some noises in the text, cosine similarity is computed based on $TF - IDF$, and the K-means cluster algorithm shows the attribute cluster. The algorithm complexity of this is $O(NlogN)$. Fourth, these clusters are matched to the profile attribute. Table 2 summarizes the proposed framework.

4. EXPERIMENTS

In order to verify our approach, we did the experiment with fifteen thousand resumes in Chinese which provided by a commercial head-hunting company. These resumes are contains different industry field people and different source.

We use precision, recall and F value to evaluate this approach. Cause the dataset is huge, the standard precision and recall can not be compute without the label data directly. A score function is involved to compute these three criterion, which is defined based on the importancy of each field in the resume as shown belows.

$$Score = 5 * importantItem + 3 * optionalItem$$

The score function treats each field of the block as a unit,

Table 2: Algorithm to extract the resume

Input: L: Set of n lines of each resume; Output: R: resume with structured data
1. for line in L if line match heuristic rules do operation 2. for line in L find pattern of line match the pattern to others if match record the block else continue return all blocks 3. for block in B match the name entities 4. return resume

Algorithm 1 Framework of extracting information from raw resume text.

```
for each line ∈ lines do
    if line match heuristic rules then
        do operation
    end if
end for
for each line ∈ lines do
    find pattern of line
    match the pattern to others
    if match then
        record the block
    else
        continue
    end if
end for
record all blocks
for each block ∈ blocks do
    match the name entities attribute
    if match then
        save the name entities
    end if
end for
```

then compute the total score of each resume.

We supposed each resume text has basic information, education, work experiences and self evaluation things. This hyperspace is not match the real data, but as a result of the huge volume it does not matter to get the basic overview. Because the extracting algorithms are independent of the test corpus, we compared our approach with PROSPECT [**?**], which also use the natural language processing method to extract the detail information from the resume text. However, PROSPECT used upwards of 7200 human annotations about the name entities and 110 English resumes segment annotations as the system cold-start resources.

Table 3 shows the result of our experiment and Table 4 shows the result of PROSPECT. From the results, we can get an overview about the resume dataset that most resume can be detected by our approach and the prcision and the recal are acceptable. To analyze the details of each part of the resume, we used the score function to evaluate the results.

Table 3: The evaluation of results

block name	prcision	recall	F value
name	0.952	0.919	0.935
email	0.992	0.714	0.830
other basic information	0.923	0.75	0.823
education	0.912	0.701	0.792
work experiences	0.873	0.720	0.789
self evaluation	0.897	0.796	0.843

Table 4: The results of PROSPECT

block name	prcision	recall	F value
education	0.940	0.902	0.921
work experiences	0.790	0.780	0.785

Basic information. From the table above, it's easy to know that most resumes contains the name and email information, which consistent with our intuition since job seekers must leave their contact information on their resumes. In addition, person name has obvious characteristics so that it's easy to detect and regconized. The email also has an obvious feature, which is contrustred by servel characters and only one @ symbol. Other basic information concludes how many years he/she worked, address, sex, id number, phone num. Most resume contains the basic information but not each of them, which inflect the recall.

Education. For education module, the score function is made up of university name, education time, major, degree and courses, where only the courses are the optional part. From the statistics in the Figure 3, most of the data nodes are around the 15 score, which shows that most people has only one record about education. Those data node that scores are above 50 are very likely to be wrong extraction result.

Compared to the PROSPECT, the gap on precision is not big, but the gap is obvious on recall. This gap caused by the data set, since the data of PROSPECT are more pure to complete resume. But the data in our experiment comes from the crawler of search engine, lots of "resume" data are not complete and some of them are notes from the headhunters.

Work experiences. For work experiences module, the score function is made up of company name, work date, job position and description which is optional part. As shown in Figure 4, most of the data within the range, between 10 and 70. In other words, most job seekers have served for less than three companies. This is very reasonable to the fact that the resumes are collected from the internet and the mainstream users of recuriting platform are less than 30 old.

Global evaluation

This is the mainly reason of the low rate of education and work experience, whose block need carefully detected.

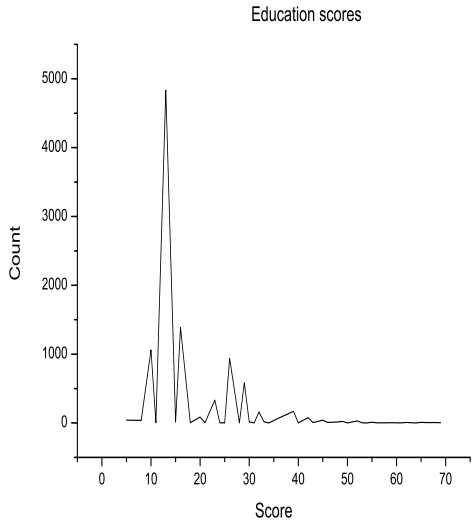

Figure 4: Entity extraction results for education block

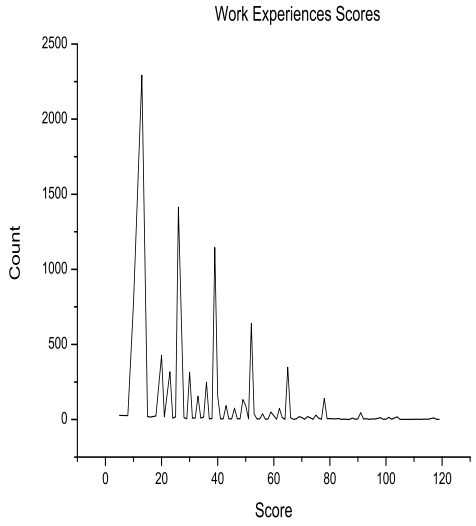

Figure 5: Entity extraction results for work experiences block

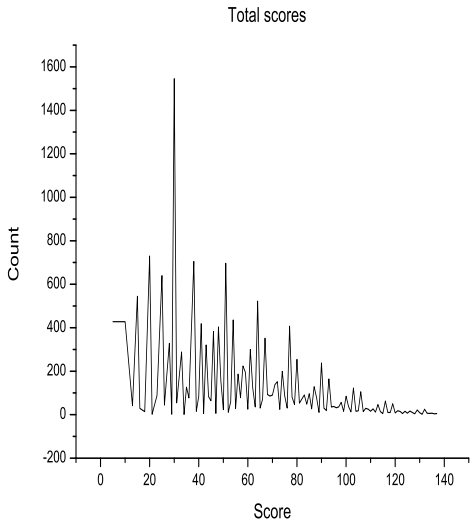

Figure 6: Entity extraction results for the whole resume

Compared to other approachs published in related works, our method is easy to implement and also gain a considerable result. Without too many human label data is another advantage.

5. CONCULSION AND FUTURE

In this paper, we propose an approach to extract the details from unstructual resume text. This work used to extract the detailed information from the raw resume text in order to improve the efficiency of reading resume for head-hunters. The most contribution of our work is that our framework can extract the details of resume without too much labeled data with simple model. The results are acceptable and can reach the state of art of solutions for this problem.

In the future, we will try to introduce our framework to English resumes and try to auto-generate the e-recuriting domain knowledge base in order to gain better extractor performance.