УДК 681.142.2

Ю. О. Герман[1], О. В. Герман[1],С. Наср[2],

[1,2]Белорусский государственный технологический университет;
[3]Белорусский государственный университет информатики и радиоэлектроники

МЕТОД ИЗВЛЕЧЕНИЯ ИНФОРМАЦИИ ИЗ РЕЗЮМЕ

Предлагается подход для извлечения информации из коротких и плохо структурированных текстовых документов, например резюме (CV). Компьютерная обработка резюме является актуальной и интересной прикладной задачей. Имеется ряд сайтов для централизованного размещения резюме, ориентированных на различных работодателей. Работодателя часто интересуют детали, а не резюме в целом. Особенно это касается профессиональных навыков и достижений заявителя. Извлечение такого рода информации уже является проблемой, если резюме составлено в произвольной форме, плохо структурировано, содержит грамматические ошибки. Предлагаемая статья именно и ориентирована на обработку такого рода заявлений. Приводится анализ существующих подходов к извлечению информации из резюме, обоснован выбор подхода, использующего ключевые слова, с помощью которых можно эффективно извлекать информацию, интересующую работодателя. Отмечена специфика подхода для случая, когда ключевые слова определяются для блоков текста с фиксированным смысловым содержанием. В этом случае возникает еще одна проблема, связанная с определением таких блоков. Предлагается подход на основе техники кластеризации, так что каждый кластер ассоциируется с соответствующим.блоком текста. Вместе с тем, техническая реализация этого подхода остается открытой и составляет предмет дальнейшего исследования. Приведены примеры, иллюстрирующие излагаемую технику извлечения текста из резюме как релевантного ответа на соответствующий запрос произвольной формы.

.

Yu. O. German[1], O. V. German[1], S. Nasr[2]

[1]Belarusian State Technological University;
[2]Belarusian State University of Informatics and Radioelectronics

INFORMATION EXTRACTION METHOD FROM RESUME

An approach to information extraction from short and poorly structured text document such as resume (CV) is suggested. The computer-based resume processing is an actual interesting application problem. There are a number of web-sites for centralized CVs allocation oriented at different employers. Oftenly, an employer is most interested in some peculiar features connected to professional achievements and knowledges of the applicant, not a resume as a whole. Extraction of such peculiar information from CV is a problem itself especially if the CV is organized in an arbitrary form, poorly structured and contains grammatic mistakes. The suggested paper is devoted to processsing the CVs of this type. A short review is given of the existing approaches to information extraction from CV and the keyword-based approach is selected and founded from the viewpoint of efficient information extraction the employer is interested in. The specificity of the approach is emphasized for the case when keywords define text blocks with a definite conceptual content. In this case one more problem arrises connected to text blocks definition. An approach based on clustering technique is suggested, so each cluster is associated with the corresponding text block in the raw CV. At the same time, the technical realization of the approach suggested remains open for future investigations. The examples are given illustrating text extraction technique to get a relevant answer to arbitrary employer query addressed to CV.

Key words: resume, search retrieval system, text processing, key words search, clusterization.

Introduction. One of the actual applied problems is automatic resume (CV) processing. There remain a lot of job applications which are unstructured or badly organized, contain grammatical mistakes or even are incomplete.

The other problem consists in potentially tremendous amount of CVs which should be processed in quite a short time interval. Finally, an employeer may be interested in some specific features the applicant should possess. It follows from this that there is need in a computer method to extract necessary data with respect to resume with poor organization, mistakes and incompleteness.

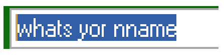
There are three main approaches to realize such a method: keywords usage [1, 2], getting a DOM-structure of the resume considered as HTML document [3] and text classification approach [4]. There still remains a necessity to cope with grammatical mistakes and provide relevant answer to the query. From this viewpoint we restrict our considerations by keywords usage approcah only with necessary modifications. Indeed, the approach which uses DOM-structure supposes that an original document is Html-document with strictly defined structure. This supposition is too restrictive for our goals because our approach allows any form of CV even with semantic discrepancies. What concerns text classification approach it also supposes that the CV text is divided into blocks with some strong semantic: one block is used as personal data including name, date and place of birth the other block stands for professional features and so on. We exclude this approach from our consideration.

Main part. Consider as an example the next CV:

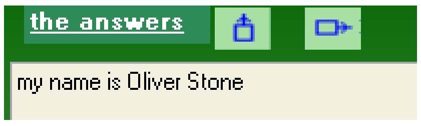
1.my name is Oliver Stone; (A)
2.i am 23 years old, unmarried; (B)
3.i am a junior researcher in university of Faradei; (C)
4.i got a BS in computer science in 2010 from Royal College; (D)
5.i graduated from Royal College at 2010; (D)
6.know such computer languages as c#, java, python; (E)
7.my interests are connected to programming and languages; (E)
8.my salary requirements are moderate; (F)
9.was engaged in some projects with practical outcome; (G)
10.i have published some articles in computer magazines in my professional areas; (G)
11.i like to read books and listen to modern music; (H)
12.also, i have vivid interests in business and financial programming; (H)
13.support good relations with other people ;(H)
14.by this I apply for a vacant position of a java programmer; (I)
15.my contact address is Belheigm city, Corwell street 10. (J)
16. please, use my e-mail: olstn_2123@cor.com (J).

In round brackets we placed the semantic block identifiers (this point is discussed later).

The computer program we developed provides a possibility to input any question to this text and get an answer. A question may contain distorted keywords, for example:

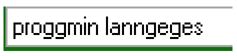

whats yor nname

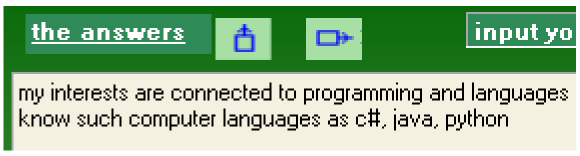
The answer is placed below



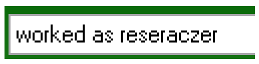
my name is Oliver Stone

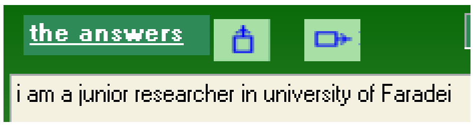
Other examples:


proggmin lanngeges



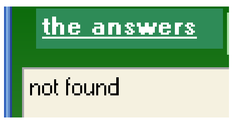
my interests are connected to programming and languages
know such computer languages as c#, java, python


worked as reseraczer



i am a junior researcher in university of Faradei

A question is formulated as a set of (key)words with possible grammar mistakes. The unswer is presented as a text block or even a single clause from resume relevant to this question. If answer is not found then we have the next resulting screenshot



not found

We have to explain the method of CV-text processing. The entire process is divided into four stages. The first stage is to transform Word document or a pdf-file with CV to a plain text. This can be done with a Tika system [5] which extracts a raw text and deletes unnecessary control information such as colors, fonts, and the like. The next step is to get keywords of the text (see [6]). The idea is to consider practically all text words as keywords due to not big size of CV. It is required that each sentence in CV contains at minimum one keyword. Besides, this simplifies the algorithm as it not requires to define each keyword score and test that each sentence is covered at least by one keyword.The prepositions, conjunctions, pronouns, auxiliary and modal verbs (such as *can*, *have*, *may* etc.) are excluded. For the cursive text of CV given as an example above, the set of keywords contains the words *name*, *Oliver*, *years*, *unmaried*, *research-*

er, *university*, *Faradei*, *College*, *interests* etc. The keywords are selected in such a way that the similar words are identified as the same. For instance, *programming* and *programmer* are considered as one keyword. By thus, each keyword labels one or more sentences. To realize the keywords selection we use the Dice metrics (measure) given by the formula

$$P = \frac{2 \cdot |X \cap Y|}{|X| + |Y|},$$

where $|X|$ ($|Y|$) stands for the set X (Y) size. For example, for X = *programming* and Y = *programmer* one has $P = 2 \cdot 8 / (11 + 10) \approx 0.8$. We adopt the rule accordingly to which two words are considered alike if the Dice measure is 0.5 or greater. From this, in the example above the words *programming* and *programmer* are considered a one keyword.

It should be noticed here that fuzzy sequential text searching methods represent evident practical interest especially if they are applied on servers with very large amount of CVs to be processed [7, 8].

The next stage consists of building two directories. Each directory represents a collection of the pairs <*key*, *value*>. The first directory contains the pairs of items and each pair represents a record with *key* standing for a keyword and a *value* representing the set of sentences numbers labelled with the keyword *key*. The second dictionary consists of the pairs representing the sentences numbers and their texts. Now let us explain how to extract an answer to the question represented by a set of words (some of words are keywords and some – not (these latter words do not belong to CV text)). For each keyword k_i in the question the set of sentences numbers $N_i = \{n_{i1}, n_{i2}, ..., n_{iz}\}$ is defined. Then for all N_i the most frequently encountered number(s) n_w is (are) defined. This number n_w defines a sentence to be displayed as an answer. If there are more than one candidate to be an answer then all candidates are displayed.

It may be the case when CV contains quite a big blocks of a semantically connected information. This situation is somewhat wider than that one considered above. Each semantically united block may consist of one or more sentences. The block specifies some concept, that is, it may be considered as a semantical whole. Practically, we associate such blocks with paragraphs (identations) or the text segments separated from the others with empty lines or by spaces. As before, we should define the keywords for the sentences: $k_1, k_2, k_3, ..., k_z$.

We introduce the third dictionary with the *keys* representing the sentences numbers and *values* standing for the text blocks. Thus, we got the chain: keyword $\rightarrow$ sentence(s) $\rightarrow$ text block(s). The searching procedure remains as before. The keywords from the query are used to find the sentences numbers and the corresponding text blocks. The text block which is referred to by the majority of keywords is then selected. To be more understandable, let us take as an example the query:

which programming languages are you interested in?

The keywords *programmer*, *language*, *interests* all are presented in the query with the corresponding sentences numbers:

programmer: 7(*E*), 12(*H*), 14(*I*)
language: 6(*E*), 7(*E*)
interests: 7(*E*), 12(*H*)

In round brackets the block identifiers are placed. As one can see, the block *E* is referred to most often and is therefore selected. It includes the sentences 6 and 7. The final step consists of definition of the text diapason to be displayed. Evidently, the rational position is to display all the sentences with the numbers starting from the minimum and ending with the maximum number among the sentences numbers selected. That is, in our case the sentences 6 and 7 will be displayed both:

know such computer languages as c#, java, python; my interests are connected to programming and languages.

Let us consider one more example of that kind. Suppose that a query is the following:

what are your interests?

Here the only keyword is presented:
interests: 7(*E*), 12(*H*)

This time the answer is composed of the 7-*th* and the 12-*th* sentences.

The problem of blocks definition has a special scientific meaning. The simplest case consists in dividing the text by paragraphs separated with indentions or empty lines. Evidently, each block should contain the connected notions (keywords). One says that related keywords form a cluster. So, one needs to build a number of clusters, combining different but semantically related keywords. The first problem with clusters is their number. Obviously, one needs to apply the existing clustering means to define optimal cluster structure. For these aims one can use a Python statistical modules such as SciLearn [9] or *R* language [10] or specialized programming packages such as [11], for example. As an alternative approach one can use a correlation-based technique [12]. The second problem is to map cluster structure to blocks. The idea is to apply covering procedure to associate the clusters with the sentences labelled (covered) with the keywords from that cluster. One, however, should keep in mind that the covering procedure must select the sentences with sequentially ordered numbers. Because this is quite a special issue going beyond the frames of the paper we omit it.

In statistics, the measure of items connectivity may be characterized with correlation coefficient [12]. The correlation coefficient between two keywords *x* and *y* is defined as below

$$r = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2} \cdot \sqrt{\sum_i (y_i - \bar{y})^2}}$$

With the help of the above formula one is in position to find pairwise correlations of the keywords. Here, $x_i = 1$ if keyword *x* occurs in the i_{th} text block (the same is right with other keyword *y*). A cluster structure we are interested in should provide maximum value of the correlation coefficient weighted through the entire CV.

Conclusion. The above given material can be directly used in automatic CV processing. There are some main application problems which can be solved with the outline technique. The most important is that one connected to extracting necessary information from the CV text which represents interest to employer. The approach we used here do not restrict the type of information. It is an evident advantage of it. The user can use mistakenly written queries and apply them to mistakenly written CVs. The second type of the problems to deal with is CV automatic structuring to write them into database. Our approch makes one enable to get structured CV by cluster definition and labelling clusters with keywords.

There remain some open questions though. The most important one is a mapping the cluster structure to blocks structure. This issue requires father considerations. In general, one may admit that the same sentence belongs to different blocks. We use an approach as described below. To define text blocks we select the verbs supposing that each verb stands for some relation(s) in the domain of the text and plays a role of some semantic function defined on the domain. The commonly used verbs (such as have, can, be, get *etc*.) should be excluded. Also it is important that total number of English vocabulary verbs is not so big and gives a possibility to use them from a previously built and indexed database. Let us address the abstract of this article for example. The set of verbs consists of the next items: *suggest* {1, 6, 10, 11}, *process* {2, 6}, *extract* {1, 5, 7, 12}, *orient at* {3}, *select* {7}, *is interested* {2,4,7}, *employ* {3, 7}, *define* {8, 9}, *connect* {4, 9}, *arise* {9}, *base* {2, 10}, *cluster* {10}, *organize* {5}, *associate* {10}, *correspond* {10}, *realize* {11}, *apply* {2, 4}, *illustrate* {12}, *address* {12}, *resume* {2, 4}, *investigate* {11}, *contain* {5}, *devote* {6}. In figure brackets the sentences numbers are placed. Our supposition is that verbs define sense of the text and so they can be used to select text blocks (as segments with some completed conception). The next step is to define a verb set covering with minimum number of verbs selected. A covering π consists of the verbs and each sentence from abstract contains at least one verb from π. For instance, one of the possible minimum-size covers is represented by set π = {*extract*, *is interested*, *define*, *employ*, *suggest*}. We should notice that finding a minimum-size covering is a hard computational complexity problem, so one can use any good heuristic method to solve it. The size of π represents the text blocks number. In our example the paper abstract should be divided into 5 blocks. The simplest way to define them is to use the numbers of the sentences covered by the verbs. For instance, verb *extract* covers the sentences 1, 5, 7, 12. So, the sentences with these numbers define a text block in the approach we describe. We have 5 blocks here: *A* = {1, 5, 7, 12}, *B* = {2, 4, 7}, *C* = {8, 9}, *D* = {3, 7}, *E* = {1, 6, 10, 11}. From this, one can see that some sentence may belong to different blocks at the same time. Consider, how a query «*extract info employer*» is treated. The parcing procedure results in

extract: 1, 5, 7, 12(*A*)
info: 1(*A*, *E*), 5(*A*), 7(*A*, *B*, *D*)
employer: 3,7(*D*)

Non-keyword item *info* refers different blocks. The mostly referenced are blocks *A* and *D*. So the answer is either *A* or *D* or both of them. Each of the blocks *A* and *D* should be scanned separately in order to define the most relevant answer. It should be noted here that additional block processing is needed as the most relevant answer may correspond even to a part of block. Thus, in the example we have

(1) *An approach to information extraction from short and poorly structured text document such as resume (CV) is suggested.*

(5) *Extraction of such peculiar information from CV is a problem itself especially if the CV is organized in an arbitrary form, poorly structured and contains grammatic mistakes.*

(7) *A short review is given of the existing approaches to information extraction from CV and the keyword-based approach is selected and founded from the viewpoint of efficient information extraction the employer is interested in.*

(12) *The examples are given illustrating text extraction technique to get a relevant answer to arbitrary employer query addressed to CV.*

From these sentences only (7) and (12) should be taken as an answer from block *A*. Consider then block *D*:

(3)There are a number of web-sites for centralized CVs allocation oriented at different employers.

(7) *A short review is given of the existing approaches to information extraction from CV and the keyword-based approach is selected and founded from the viewpoint of efficient information extraction the employer is interested in.*

Sentence 7 is the most relevant. Combining both blocks *A* and *D* the final answer is represented by the sentence 7.

Evidently, one can apply the total approach not to CV only. Any type of not long texts can be processed in the same way.These include the paper abstarcts, *e*-mail contents, patent formula descriptions and the other short document types. As a further research direction we point to short document clustering problem. The idea to use keywords is not the only one. There also may be used text ontology tools, semantic networks and the other means. It is also necessary to consider usage of synonyms.

References

1. Maheshwari S., Sainani P., Reddy K. An approach to extract special skills to improve the performance of resume selection. In: Databases in Networked InformationSystems. Lecture Notes in Computer Science. Vol. 5999. pp. 256–273.Berlin, Germany, Springer, 2010.

2. Kopparapu,S.K. Automatic extraction of usable information from unstructured resumes to aid search. Proceedings of the 1st IEEE International Conference "Progress in Informatics and Computing (PIC '10)". 2010, vol. 1, p.p. 99–103.

3. X Ji, Zeng J, Zhang S, Wu C. Tagtreetemplate for Web information and schema extraction. Expert Systems with Applications. 2010. vol. 37, no.12, pp.8492–8498.

4. Yu K, Guan G, Zhou M. Resume information extraction with cascaded hybrid model. In Proceedings of the 43rd Annual Meeting "Association for Computational Linguistics (ACL'05)". 2005 June. pp. 499–506.

5. The Apache Sofrware foundation. Apache Tika – a content analysis toolkit. Available at http://tika.apache.org (accessed 12.02.2019).

6. Buttcher S., Clarke C. L. A., Cormack G.V. Information retrieval. Implementing and evaluating search engines. The Mit Press. London. England. 2010. 606p.

7. Tarhio J., Ukkonen E. Approximate Boyer-Moore String matching. SIAM Computing. Vol.22. 1993. p.p.243–260.

8. Anu S., Joby G. Fuzzy pattern matching algorithm for location based approximate strings. International journal of scientific and engineering research. Vol.7. Issue 7. 2016. p.p. 583–587.

9. Sheppard K. Introduction to Python for econometrics, statistics and data analysis. University of Oxford. 2014. 394p.

10. Trevor M. Undergraduate Guide to R. Available at http://www.biostat.jhsph.edu/~ajaffe/docs/undergradguidetoR.pdf (accessed 12.02.2019).

11. Kirilov A. Projects and applications using AForge.NET Framework. Available at https://www.codeproject.com/Articles/16859/AForge-NET-open-source-framework (accessed 12.02.2019).

12. Cox D.R., Snell E.J. Applied statistics. Principles and examples. Chapman & Hall.1981. 190p.