Second Asia Pacific Conference on Contemporary Research (APCCR, Malaysia, 2016)
ISBN:978 0 994365 606
www.apiar.org.au

INTELLIGENT RECRUITMENT MANAGEMENT ENGINE

Chiththananda H.K.I.C.L. [a], Perera T.D. [b], Rathnayake L.M. [c], Mahanama M.G.G.D.D.P. [d], Prabashana P.M.P. [e], Dias D.P. N.P. [f]
[abcdef] Sri Lanka Institute of Information Technology, Sri Lanka

Abstract

A system which can be used to automate the recruitment process of an organization is proposed in this paper. The system is designed to target the human resource department of an organization in order to simplify the massive process of data extraction from a large number of curriculum vitaes (CVs), on the other hand to reduce the cost and time which have to be spent on the interview process and to allocate most suitable interviewers for each interview by analyzing available past data. Information extraction (IE) is used in order to retrieve data from CVs and cover letters. An ontology map is created to analyze and categorize the extracted key words through this system. Then, the CVs are sorted and prioritized according to the given requirements of the organization. In addition, prediction components have been developed and embedded in the main system to predict the future cost in recruitment operations of a particular organization. Hence, it is believed that this system will enhance the efficiency and effectiveness of the recruitment process of organizations.

Keywords: Information Extraction, Information Optimization, Ontology, Prediction

1.0 Introduction

Recruitment is one of the critical processes of an organization's human resource (HR) planning. The Recruitment process in every organization is becoming complicated due to elevation in global competitiveness and raise in labor market flexibility. Presently, most of the organizations are paying more attention to conduct their organization's recruitment process with low budget and minimum resources, with the target of having the right person, in the right place, at the right time. However, most of the time, this ultimate goal of recruiting is improbable due to several of reasons pop out in resource management such as having right interviewers, venue, cost etc. Nearly all companies are following the traditional ways of the recruitment process by going through all the applied CVs manually and filter candidates for interviews. Sometimes, some CVs may get ignored by the responsible persons mistakenly which will make a long term lost to the company.

After diagnosing the above issues, we arrive at a decision, that it is possible to automate this process in a more accurate way. Finally propose and implement a system, "RecMan" which is an intelligent recruitment management engine(IRME)where this particular stage of the recruitment process will be automated for the beneficial of the company. This automated engine will help the company to reduce the cost and the time which they have to spend unnecessarily on these steps and again the HRM has the full freedom to include his/her time for another task.CV shortlisting plays a major role in RecMan, though there are different CV formats, IRMEwill use keyword identification without any impediment.

1.1 Contribution

Unnecessary spending of money and time will be reduced by the Recman as a result of automating and it is often possible to do CV filtering mechanism manually by the HR manager according to company requirements, policies, etc. RecMan will handle all types of different CV formats (most of the time unique for individuals) using key words identification without any impediment. Forecasting is another value added feature of this system which provides predictions, suggestions to improve quality of human resource planning and to help make organizational strategic decisions. RecMan will help to improve the quality and accuracy of organizational recruitment process, in aspects of time, resources and cost. The main contributions of the paper are: For retrieving data from CVs, IE is used along with optimization. Information will be analyzed using NLP and XML based dictionary to get the most suitable CVs.

1.2 Related work

Academic and commercial world's literature includes a variety of techniques and tools to provide assistance in recruitment process of organizations. "KnowRex" is such a system which used ontology-driven information extraction [1]. However, some bottlenecks were found in that existing system. The main bottleneck in "KnowRex" is, it only supports the Europass style CVs which also known as two dimensional CVs.

According to the previous research, this Ontology-driven Information Extraction (Weronika & Adrian, 2015) was done by using Ontology, text annotation, information extraction and table recognition. The researchers were able to develop a system for Europass CVs to offer Semantic view of it. In KnowRex, the whole process was divided into two phases: design and runtime. In the design phase, they are defining the schema for the semantic view, fixes objects, arrange the annotations, partitioning unstructured document and provide rules. In runtime phase, the system processes the documents.

In the above research, several APIs are used in order to come with their product such as StanfordNER, Alchemy, DBpedia Spotlight, Extractiv, OpenCalais, Lupedia. This product is named as KnowRex (Weronika & Adrian, 2015) which is a framework allowing the users to describe the final semantic view and is independent from the basic structure. KnowRex has used bi-dimensional processor, one and bi-dimensional tokenizes, annotators, semantic descriptors and logical rules as tools and the techniques. For the extraction step the bi-dimensional processor, annotators and descriptors must adjust and for mapping logical rules must be used.

2.0 OVERVIEW AND DESIGN

2.1 Information Extraction (IE)

In our research the information extraction component is mainly developed in a Linux environment using python. Python is a high level language and its execution speed is considerably higher than the other languages. The team was able to write less number of lines of code than the other languages to come up with the algorithm. Since UI (front end) has developed in .NET platform the algorithms have written as a web service in python.

Initially, the system converted the CVs' data (PDF) into a machine readable content. For this, we have used textract package in python. Textract[13] is used to extract text from any kind of a document. The "process" method in textract library has been used to ensure that all text from the uploaded CVs were extracted. After extracting the text the algorithm splits it into data elements using new line character("\n") and stores it in a separate array. So, according to a given format , it'susing the textract library to extract the basic details of the CV such as Full name, Address, Contact No, Date of Birth, Email, Gender, Marital status, Nationality, Spoken languages etc. Those are the basic details of the applicant. To get those details algorithm was made out by considering the format of those details mentioned in the particular CV. Considering those keywords IRME is taking the text in front of those keywords to do the basic extraction.

Here, our main focus is on an IT industry related CV. So, the applicant mentions about his/her key skills in the CV. Considering this factor and the format of the CV,IRME came up a way to extract the key skills of the applicant. He/she mentions them under key skills and competencies section of the CV with keywords like computer skills and database. Using a while loop, some if conditions and with the help of regular expressions in python finally able to extract those technical skills of the applicant and those details will also append to the user object that is parsing through a JSON array.

Considering the particular CV format, we can see that the projects which are done by the applicants are mentioned under Projects and Research section with some descriptions of them. To extract those information in the descriptions the NLTK (Natural Language Toolkit) library has been used in python.

NLTK is a platform where it eases to work with human language data. It provides many text processing libraries such as classification, stemming, tokenization tagging, parsing etc. Since NLTK is an open source package, we did some tests to come up with a way to retrieve keywords from the project descriptions. While working on those, another package was found called Rapid Automatic Keyword Extraction (RAKE).

RAKE consists with an algorithm which can be used to automatically extract keywords from documents. This package also follows natural language processing techniques. So, to work with this library, we need to have the NLTK library also included in the project. RAKE depends on the NLTK package because it uses some libraries which are included in NLTK (RAKE, n.d.).

The main task of RAKE is checking the frequency of words and it allows us to define a threshold based on the frequency of the key words. It defines whether the certain keywords are important or not based on the frequency of them appearing in the project description. This can eliminate all the stop words and annotations in a given text. It checks the phrases and whether the adjectives are matching or not. The important thing about this RAKE library is that if taking the stop words list, it can manually maintain a stop words list as well as it checks the stop words from the NLTK libraries. It finds both the lists and the frequency of appearing the given words in a corpora. This can give a keyword list separately. Once rake is analyzing a text, it looks for the frequency of a particular word or phrase. If it's there in the list there is higher chance of that particular word or phrase to be a keyword. The project details come with a summarized version of it, which means that it takes all the necessary keywords from the project descriptions which can be used later in the shortlisting component in the product. We have used .NET and Java Compatible library which can do the extraction & optimization task step by step. Since our main backend is in .NET platform, ITextSharp (Textract, n.d.) in .Net Library was selected. It can assemble, expand, extract, split and interact with any PDF file. It is compliant with most ISO PDF standards. For the extraction purposes, the extraction component is used. According to their API Document, we have first created PdfReader object, because our main input file type is in PDF. Then, parsed the CV pdf path into the reader and completed the reading process with that. Next to extract text using an extraction strategy from the API. For that PdfTextExtractor class is used. API'sgetTextFromPage method was selected to implement the method. PdfTextExtractor can parse read page, number of pages and the strategy name to do the process. In interface ITextExtractionStrategy, provided us extraction methodologies named SimpleTextExtractionStrategy, LocationTextExtractionStrategy (Simple Text Extraction, n.d.(a); Simple text extraction, n.d.(b)).

A text extraction renderer keeps track of relative position of text on page. The resultant text will be relatively consistent with the physical layout that most PDF files have on screen. This renderer keeps track of the orientation and distance (both perpendicular and parallel) to the unit vector of the orientation. Text is ordered by orientation, then perpendicular, then parallel distance. Text with the same perpendicular distance, but different parallel distance is treated as being on the same line. This renderer uses a simple strategy based on the font metrics to determine if a blank space should be inserted into the output (Simple text extraction, n.d. (a)).

For the skill matching purpose, an xml mapped dictionary was used to identify the skills. The xml map has all the skills with separate tags. We have iterated through this xml map and matched from previously extracted words.

The degrees have stored in the xml dictionary and used to find categories. Here, the special case is, it can identify both degree title and its order based on its value. We have defined degree order by assigning an order value to our XML file. Thus, it's easy to add values for our xml dictionary in order to expand the accuracy.
For ontology creation, there are lots of tools available. Among them, protégé was selected which was developed by the Stanford University. This tool uses the language called OWL language. Different ontology languages provide different facilities. The reason we have selected the OWL language is because complex concepts can be built up in definitions out of the simple concepts as it's based on different logical model.

But, the environment changed to NetBeans later because of the different requirements that the clients have moved on with this it is easy to manage with database. While searching, we found two libraries to use in NetBeans to work with ontology called:

1. owlapi-3.2.3
2. owlapi-distribution-4.1.4-javadoc

These two can perfectly be used to build OWL file. The process is as follows:

All the extracted words must be collected into a database in an order. As moving on from format to a format, we tried to make it identifiable with the basic information. So, MySQL database was created as the array with the extracted keywords list that comes into the tables. Then, it was saved in the tables. When need to create ontology to identify the relationship of they have mentioned, we called a method to get all the information. Then took word by word from the columns and made OWL individuals.

Then, we extracted to find out the relationships. Next, the keywords were mapped with the relations in the OWL object property. Finally, we were able to build the OWL file with all the entities and relationships.

2.2 Forecasting

Our main prediction is how longer an employee will stay in the company. When to resign from a company is absolutely a personal decision and most of the times, it is a secret until the particular employer hands over the resigning letter to the management. In this prediction, what we exactly try to do is identify some variables to say how longer an employee will stay in this company. We analyzed the past data of the employees to find a pattern that combines with the resignation, then decide that seventeen variables may have some relation to the employee resignation.

Those variables are gender, age, educational qualifications, designation, working experience, salary, how long an employee works in the current company, number of increments an employee get, increment rate, promotion gap, working hours per day, do they need to work on weekends, if yes do they additionally pay for that, do they have night shifts, is it flexible to take leaves, distance between the residence and the workplace, reason to stay in the current company. Then, IRME goes through the analyzing process for each variable separately to identify a pattern. From that mathematical analyze, system was able to find four variables that have a proper relation with the prediction. Those variables are salary, age, experience and the distance to the workplace from the residence. So, we filtered those four columns and do SPSS analysis to build the formula. IRME uses few SPSS mathematical analyzing techniques for analyzing these data. First, we try to find the correlations, then the ANOVA calculation and the coefficient values.

Formula
$Y = 1.931 + (1.53*10\text{-}5*S) + (0.169*A) – (0.105*E) – (0.181*D)$
Y: - How long will employee stay (1-10 years), 1.931: - Constant, S: - Salary, A: - Age,
E: - Experience, D: - Distance to the company.

From the model summary, R square value is 0.758. The mathematical and static analysts says that if the R square value is around 0.7 of data model then the data model is properly matched with the dependent variable.

The next prediction is how much an applicant is compatible with the company. What we just try to do is to get the personal qualities of an applicant and try to find whether he/she is compatible with the company. Trying to predict someone's personal qualities from a CV is an almost impossible task. We built a questionnaire and the applicants will answer for those questions in the interview time.

From the company database and the extracted information from the CVs, we are going to predict the number of CVs that will retrieve to the company in future. Then, the system has applied the Moving average method to each district separately and going to predict how many CVs will receive in the next few years from each selected district.

3.0 Conclusion & Future Work

The system is designed in targeting the HR department of an organization in order to simplify the massive process of data extraction from a large number of curriculum vitas (CVs) and to reduce the cost and time which have to be spent on the interview process and to allocate most suitable interviewers from the organization for each interviews. We did information extraction to retrieve data from applied documents. An ontology map was created for the data extraction process so that retrieved information will be analyzed and categorized through this system. Then, the CVs are sorted and prioritized according to the requirements of the organization. We came up with a prediction component which will embedded in the main system to predict the future of the incoming values such as the cost and time of the recruitment process of a particular organization by analyzing the past records.

Currently, we have done this research for some selected number of CVs. But, our target was to cover any kind of CV formats. So, one of our future plans will move on with adopting this engine for all kind of CV formats and we have planned to extract the information not only from the CVs but from the cover letters too.

The developed ontology tree was currently built to move on with English key words, but the team has a plan to develop this in Sinhala words as well, with the help of Sinhala information extraction. We will move on with different platforms to develop the ontology.

Predicting the future employee vacancies and modifying the built formula avoiding the co-relations of independent variables are some other future projects which we have planned to do in our forecasting component.

References

i. Badrul Sarwar, G.K. (n.d.) *Item-Based Collaborative Filtering Recommendation Algorithms*. Minneapolis: GroupLens Research Group/Army HPC Research Center, Department of Computer Science and Engineering, University of Minnesota, Minneapolis.

ii. Gabor Angeli, M.J. (n.d.) 'Leveraging Linguistic Structure for Open Domain Information.' Department of Computer Science, Stanford University.

iii. Goldman, R. & Widom, J. (1997) 'Dataguides: Enabling query formulation and optimization in semi structured databases.' In *Proceedings of the Twenty-third International Conference on Very Large Data Bases*, pp. 436–445. Athens, Greece.

iv. Group, M.O. (2009) *Improving the integrity of Identity Data Matching, Better Practice Guidelines*. Commonwealth of Australia.

v. Ilianna Kollia, B.G. (n.d.) 'SPARQL Query Answering over OWL Ontologies.' Oxford University Computing Laboratory, UK.

vi. Jiaze Chen, L.G. (2016). 'Information Extraction from Resume Documents in PDF Format.' Beijing, P.R. China: Institute of Computer Science and Technology of Peking University; Graduate School of Informatics, Faculty of Informatics, Shizuoka University.

vii. Joonseok Lee, M.S. (2012) *A Comparative Study of Collaborative Filtering Algorithms*.

viii. Kavi Mahesh, S.N, (n.d.) *A Situated Ontology for Practical NLP*. Las Cruces, USA: Computing Research Laboratory, New Mexico State University.

ix. Li, A.M. (n.d.) 'Early Results for Named Entity Recognition with Conditional Random Fields.' *Feature Induction and Web-Enhanced Lexicons*. Amherst, MA: Department of Computer Science, University of Massachusetts Amherst.

x. Manning Yan, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard S.J. & David McClosky (2014). 'The Stanford Core NLP Natural Language Processing Toolkit.' In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55-60.

xi. Martin Labsky, V.S. (n.d.) 'Information Extraction Based on Extraction Ontologies: Design, Deployment and Evaluation.' University of Economics, Prague, Dept. Information and Knowledge Engineering,.

xii. Pierre Baldi, S.B. (n.d.) 'Accessing the accuracy of prediction algorithms for classification: an overview.' Department of Information and computer science, University of California, USA; Center of Biological Sequence Analysis, The Technical University of Denmark, USA.

xiii. RAKE (n.d.) 'The rake package.' [Online] Available at: https://hackage.haskell.org/package/rake/. Accessed August 28th, 2016.

xiv. Sandkuhl, K.H. (2005-2009) *The State of Ontology Pattern Research: A Systematic Review of ISWC*. Jonkoping, Sweden: School of Engineering at Jonkoping University.

xv. Schumacher, S. *Probabilistic Versus Deterministic Data Matching: Making an Accurate*

xvi. Simple text extraction (n.d.) 'ITEXT.' [Online] Available at: http://developers.itextpdf.com/.../com.itextpdf.text.pdf.pars... Accessed August 20th, 2016.

xvii. Simple text extraction (n.d.) 'ITEXT'. [Online] Available at: http://developers.itextpdf.com/com.itextpdf.text.pdf.pars... Accessed August 21st, 2016.

xviii. Textract (n.d.) 'Textract.' [Online] Available at: http://textract.readthedocs.io/en/latest/. Accessed August 28th 2016.

xix. Weronika, N.L., & Adrian, T. (2015). 'Ontology-driven Information Extraction. Department of Mathematics and Computer Science.' University of Calabria, Italy; AGH University of Science and Technology, Krakow, Poland.