

Best Fit Resume Predictor

Sujit Amin¹, Nikita Jayakar², M. Kiruthika³, Ambarish Gurjar⁴

¹U.G Student, Fr. C. Rodrigues Institute of Technology, Vashi, Navi Mumbai, India.

²U.G Student, Fr. C. Rodrigues Institute of Technology, Vashi, Navi Mumbai, India.

³Associate Professor, Fr. C. Rodrigues Institute of Technology, Vashi, Navi Mumbai, India.

⁴Benco Solutions, Waghle Estate, Thane, India.

Abstract - This paper focuses on the solution developed in the form of a web application to predict the best fit resumes against a given job description posted by a job recruiter. In this prototype, the web application can intelligently predict which resumes are better fit against the given job listing based on key factors of any candidate. These key factors include, but not limited to, education, number of years of experience and skills. This solution was developed on the purpose of significantly reducing the workload of the recruiters of any company who otherwise experience the pain of manually going through the details of each and every candidate's resume from the given pool of prospective candidates. The output of this will be visible only to the recruiter in the form of a ranklist of all the candidate based on the overall resume scores assigned to each and every applicant on the basis of their education, work experience etc.

Key Words: Natural Language Processing (NLP); NER (Named Entity Recognition); SpaCy; TFIDF; cosine similarity.

1. INTRODUCTION

In today's life, thousands of people are constantly in search of jobs matching their passion and interest in order to make a living.[1] Hence, job recruitment websites are gaining popularity for the recruitment team in companies as well as the job applicants.[2] With the burgeoning population on the hunt of comfortable cushy and preferably white-collar jobs, the number of job applications for any job listing is breaking the record of crossing insane numbers.

This situation is proving to be insanely grueling and difficult for the job recruitment teams to fish out the best prospective employees for their type of job listing. The situation becomes dire when around 75-80% of the job applications submitted to the recruitment team via the web portals fail to match the job requirements as per the posted job listing specifications set by the recruiters. It is similar to the situation of finding a needle a haystack full of job applicants. Manually going through each and every resume is very monotonous for the recruiter. It is a very tedious and strenuous task for the job recruiter to manually go through each and every resume. [3] This often can lead to delay in the hiring process of the prospective job applicant.

This harrowing experience for the job recruiter can be pacified by developing a solution which will ease the workload shouldered by them. Not only will developing such kind of solution ease the workload of the job recruiters, but it will reduce the time taken typically for the hiring process to be completed. Another positive of having such a solution in the market is that it will keep the job recruiter more motivated to hire the best talent out of the huge pool of candidates in a shorter span of time easily.

This paper majorly focuses on the solution developed to address the problem faced by the job recruiters in selecting the best talented prospective employees. A web application which uses advanced AI techniques like NLP framework for evaluating candidates' resumes against a given job profile or description as well as generating a ranklist only visible to the recruiter has been discussed in detail in the sections given below. Certain important factors like education, work experience etc. should be given how much weightage is also a food for thought which has been analyzed in the sections below in this paper. There were multitudes of ways to create this web application especially on the tipping point of which method to do comparisons between the resume and job description. Cosine similarity was one of the smartest and easiest methods with less time complexity used for this purpose and it has also been discussed below.

2. LITERATURE REVIEW

There are a plethora of websites and web applications developed by a myriad of developers in order to address parts of the tedious recruitment process problems. There are over 50,000 websites which have developed till date. Some of these are Adecco.com, Monster.com, Top resume, Ideal etc. More than half of these websites have overlooked the problem faced by the job recruiters to go through each and every resume manually in a tedious manner.

Let us take two cases of websites developed for job recruitment. The first website taken as a case study for understanding the drawbacks of it is Indeed. Indeed is a very popular website used by a lot of companies, recruiters and job applicants. It employs the facility for the job applicant to upload their resume and apply for multitudes of job listings available on the website. They can apply to more than one type of jobs. The employers or the job recruiters also have the option to post one or more job listings on the website. Thousands of applicants apply to a myriad of job postings open on the website. The problem here is that these recruiters have to do a monotonous job of going through each and every resume to find the best fit candidates out of the big pool of prospective job applicants. No provision is made to solve this issue of the job recruiters. Some other websites like glassdoor allows additional facilities such as rating and reviewing the job listing. This too fails to solve the problem faced by the recruiters.

In another case study, let us take the website Top resume. This website employs the candidates as well as the job recruiters to upload the resumes they have in hand and get the rating of the strength of the resume. Although this tool is extremely helpful for the job applicants to assess the strength of their resume, it is not at all helpful for the recruiters since they still have to upload each and every resume to assess the overall strength of the resume. Another drawback in this case is that the resume cannot be compared for a given job profile or listing. Hence, no ranklist of the best fit candidate profiles can be generated out of the given large pool.

3. DESIGN

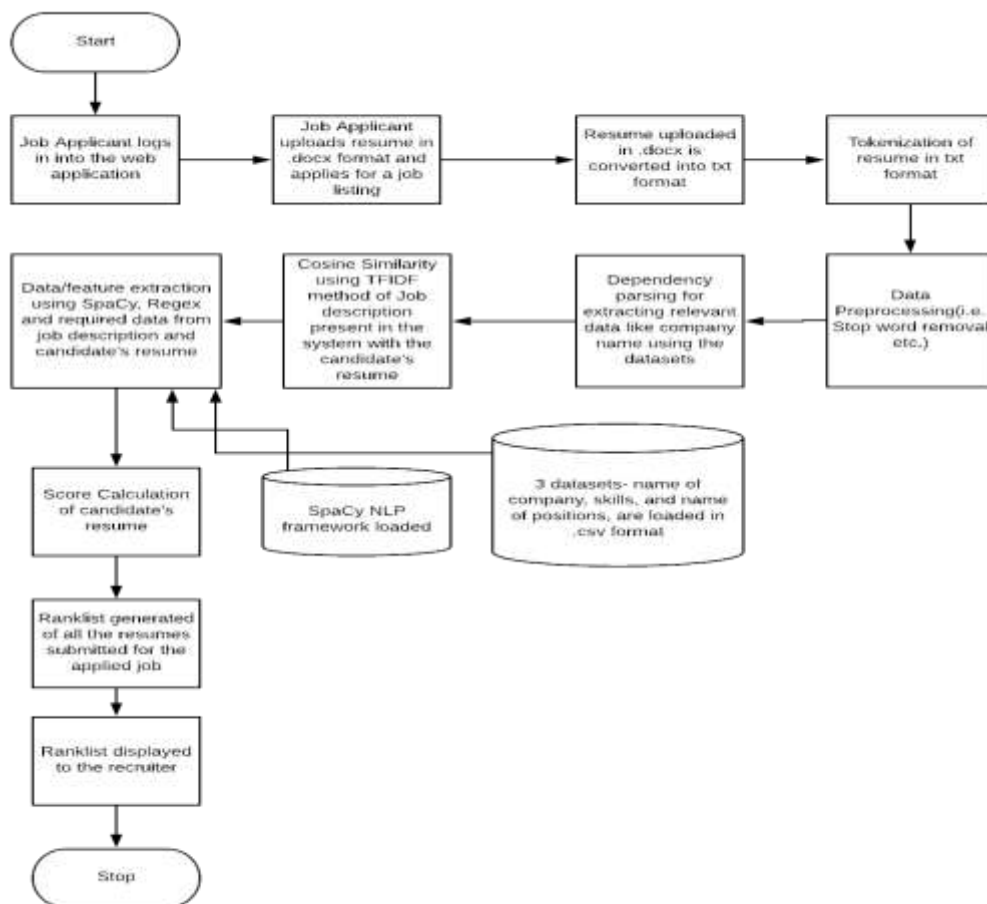


Fig -1: Workflow of the entire application

Fig. 1 depicts the overall workflow of the Natural Language Processing Model. The web application for screening of the resumes has been carefully designed in such a way that there are two types of users, i.e., the candidate and the recruiter. The entire workflow of the web application is a primitive model to demonstrate the prediction of the ranklist of the submitted resumes against the job description. Now, let us understand the in-depth details of the working of this web application by getting the insights of what is happening at both the types of users as mentioned above.

3.1 Candidate side

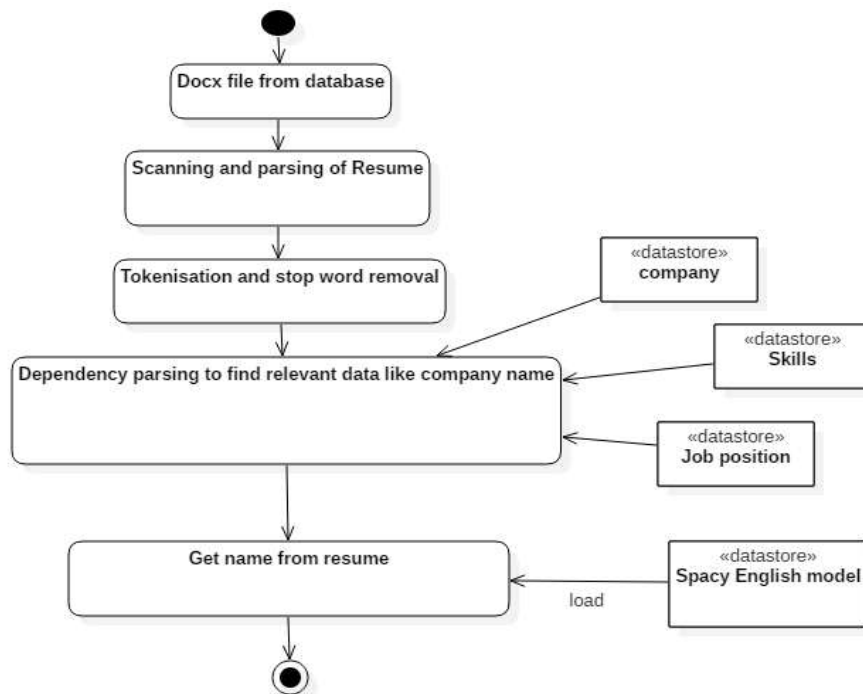


Fig -2: Activity diagram of resume acceptance into the system

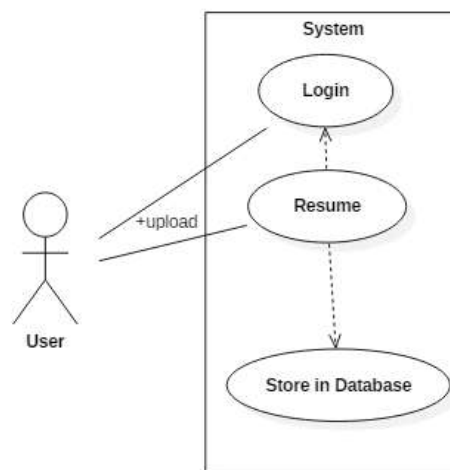


Fig -3: Use case diagram for the user as a job applicant

At the candidate's side, the candidate fulfils the initial step of feeding the resume input for the given job description. In this primitive model, there will only be one job description fed into the system as per the recruiter's requirements. Feeding the resume input is a very easy step for the job applicant. All they have to make sure is to create their account on this web application and upload their resume on the homepage after they log into the system. The job applicant's details including their resume will directly be saved in the database of the system. As soon as the job applicant clicks on submit, the data extraction using Natural Language Processing starts on the resume they submitted. This resume is raw and unstructured data and it is a challenge to extract relevant important data.

To keep up with the simplicity of the working of this web application, resumes in only .docx format are accepted for further data preprocessing. Initially, for the ease of scanning and parsing through the resume, the resume in .docx format is converted into .txt format.

Tokenization is then performed on this resume text file. After this, the resume in this text format undergoes data preprocessing. Hence, all the unnecessary stop words are truncated from the resume. For further processing, dependency parsing and Natural Language Processing using SpaCy are used to extract only that information which is relevant and vital for job matching. SpaCy uses the concept of NER(Named Entity Recognition) to identify categories such as name, company name etc. from raw and unstructured data. In order to achieve this, three datasets in .csv format are loaded into the system. These datasets have a catalog of job positions, company names and skills in job position dataset, company dataset and skills dataset respectively.

Since tokenization has already been performed on the resume, it is extremely essential to use dependency parsing for the relevant data extraction. In this concept, neighboring words are also considered along with the main word for relevant data extraction with the help of the datasets.

For example, to extract important data from the text like business technology analyst as a job position, when the parser lands on the word business, it will look for words before as well as after this word before comparing whether this group of words exists in the catalog of numerous job positions.

To extract the name from the resume, the 'en' model, i.e, the English model of SpaCy framework of NLP is used. The name is extracted accurately after this framework is loaded into the system. Finally, contact details of the candidate, such as phone number, email id etc. is extracted using Regex.

Fig.2 is an activity diagram which depicts the workflow of the above explanation regarding resumes. Fig. 3 is a use case diagram which depicts the interaction of the user who is a job applicant with the system, in this case a web application.

3.2 Recruiter side

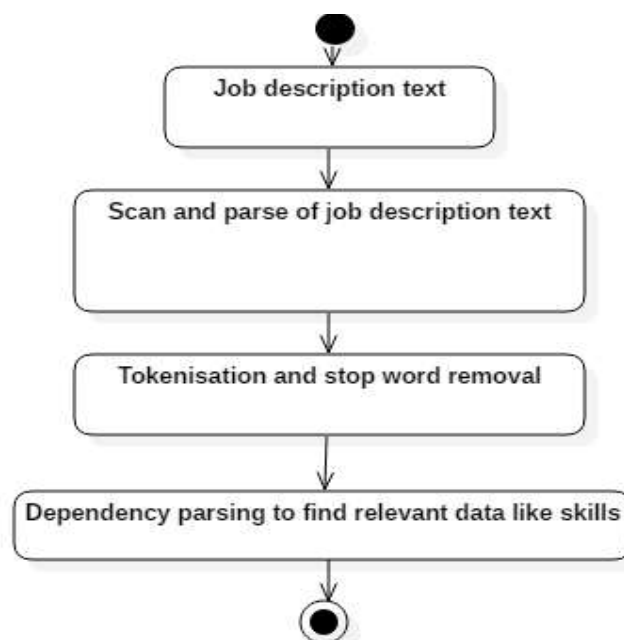


Fig -4: Activity diagram of data extraction from job description

It is imperative to understand what happens at the recruiter's side too. For keeping up the simplicity with this NLP model, there is only one description which is allowed for one job recruiter to be provided as input. The job description, just like any candidate's resume, has to go through scanning and parsing to only capture important data out of it. Here also tokenization, data preprocessing like stop word removal and dependency parsing is performed. Fig 4 depicts the extraction of relevant data from the job description.

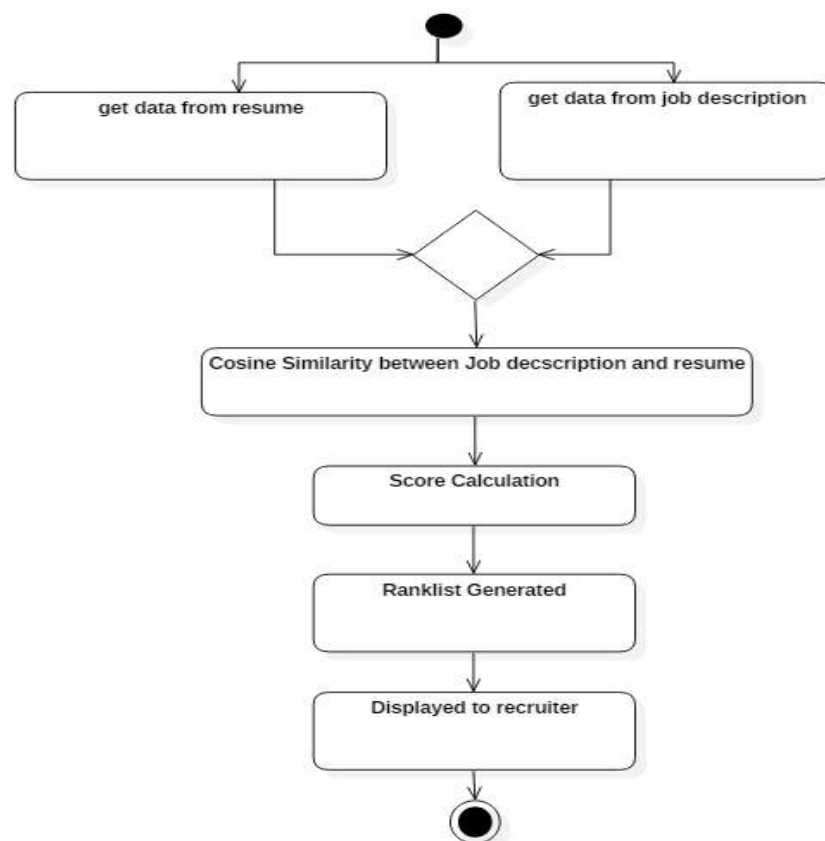


Fig -5: Activity diagram depicting the output at the recruiter end

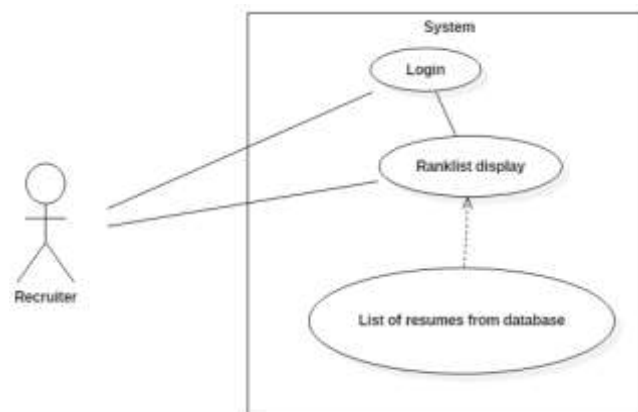


Fig -6: Use case diagram for the user as a recruiter

According to Fig.5, a comparison is made between a candidate's resume against the given job description. Comparison between these two data is done using TFIDF cosine similarity. The cosine similarity score is a value between 0 and 1, both inclusive, which resembles the cosine distance between the two data documents. Other factors which are considered for the overall resume score are education, skills and the number of years of relevant experience. Every factor is associated with a certain weightage in the calculation for the overall score of the resume. The percentage of weightage of each factor may be adjusted according to the requirements of the company using the web application. For simplicity, every factor is associated with a percent weightage by default for the resume score calculation. The score can be calculated for each resume on the basis of the following formula:-

$$S = Sr*50\% + Er*20\% + Xr*30\% + CS*10\% \quad \dots\dots (1)$$

Where,

- Sr: set of candidate's skills
- Er: set of concepts that describes the candidate's educational information
- Xr: candidate's experience
- CS : cosine similarity[1,6]

Using the above formula, each resume submitted by every applicant undergoes the same process of score calculation against the job description. Note that the score will only be calculated once the candidate selects- 'apply' to the job posting provided by the job recruiter. This has to be done as a crucial part of the checklist by the candidate on their homepage after uploading their resume. A dedicated webpage has been created for one job recruiter. On this webpage, the ranklist of thousands of candidate resumes against one job opening depending on the likelihood of the job requirements matching to that of thousands of candidates is displayed for the recruiter's reference for further rounds. An important point to be considered here is that this ranklist based on the overall resume scores for thousands of applicants is visible only to the recruiter and not to the applicants. Fig 6 provides a use case diagrammatic explanation of the interaction of a single job recruiter with the system, in this case of a web application.

4. IMPLEMENTATION DETAILS

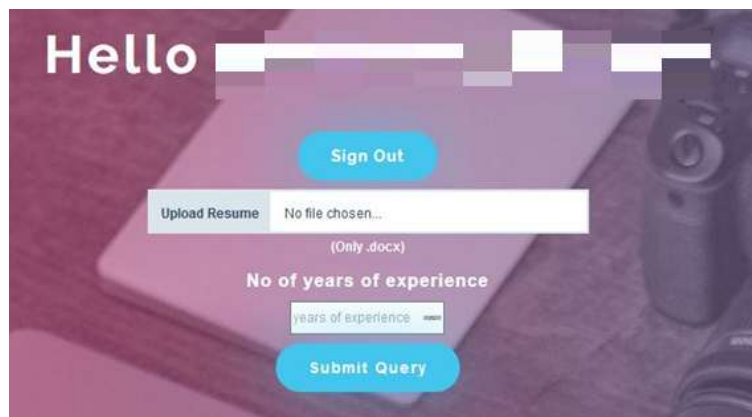
Please Sign In



Sign In

Don't have an account? [Sign Up Here!](#)

Fig -7: Sign In Page for Job Applicants



Hello [User Name]

[Sign Out](#)

Upload Resume No file chosen...
(Only .docx)

No of years of experience
years of experience

[Submit Query](#)

Fig -8: Homepage/dashboard for Job Applicants



Hello recruiter !

Job_description Ranklist

Number of applicants: 3

Rank	Applicant Name	Contact	Email	Score
1	Supriya	9987892291	supriya@gmail.com	57.25
2	Rishika Verma, Android Developer	9920625830	rishika.verma@gmail.com	34.91
3	Arundhan Kandaswamy	No Phone Number	No Email	16.25

Fig -9: Recruiter page displaying the ranklist

The above are some of the screenshots of the web application created for the ease of a single job recruiter. The technology stack used for developing this web application was the vanilla usage of the typical front-end languages, which are HTML, CSS, JavaScript and Bootstrap. These made the front-end appealing to all types of users with an attractive set of webpages. Jinja2 was used as the middleware and Flask used for server-side scripting. The AI module consisting of NLP framework commands were scripted using Python. The NLP framework used for the web application for data extraction was SpaCy English model. The datasets used for dependency parsing on every candidate's resume were in the CSV format. The database which was used to store information of the job applicants including their resumes was MySQL. The accuracy achieved for the NLP model for this web application was around 67%. For the simplicity of the web application, MySQL database was used. Hence, only one job recruiter details were stored in this prototype. The model of SDLC used for developing this web application was a hybrid of the Waterfall model and the Agile model.

In the implementation of the web application for this problem of the job recruiters, a signup as well as login has been developed of the job applicants. Meanwhile, a default separate webpage with the credentials has been created for a single job recruiter. Once the job applicant signs up to the web application, they can login into the system. They will then have the option to choose a file only in DOCX format from their local machine.

This file can be uploaded and then submitted to the system. If any other type of file format is used, then it will throw up an error on the homepage stating that the file format is not acceptable. This system was designed to accept only DOCX format. Generally, extraction is relatively easier from a DOCX document format as compared to any other format. Before clicking the button 'Submit Query', it is essential for the candidate to manually input the years of relevant experience. This action will create a record of the candidate's additional details in the database including the years of experience. The resume in DOCX format will also be stored in the database. This resume will then undergo data preprocessing and any important data will also be extracted. There were many ways in which extraction of relevant data was tried out. Undoubtedly, the easiest way of relevant information extraction was coding in Python. R could also be used but changes will have to be made in the technology stack. These days, many data mining tools have also come up in the market. RapidMiner is one such tool which was used for text extraction. Unfortunately, it was a tedious task to integrate the output obtained by this tool with the current technology stack.

For the relevant data extraction from the resume file, three datasets were self-created. These datasets were that of the company names, positions, and skills. The datasets had to be self-created. This is because especially in case of the catalog of company names, the SpaCy NLP framework model doesn't recognize the names of new companies. It only is able to recognize well-established companies like the FANG along with many others. For testing this system, thousands of open access resumes from Indeed and Google search results were considered. Many resumes out of this lot had startups as company names. CSV format was very convenient for scanning, parsing and relevant data extraction. After the text extraction of relevant data is completed, the TXT file is stored back in the database.

The recruiter is able to post one job description on the web application which is available for all the prospective candidates to apply. It is also stored in the database. The job description also undergoes similar relevant data extraction. The TXT file of this job description is stored back into the database. In the next step, the candidate is supposed to go to the job description click on 'apply' to apply for the given available job listing. Once the candidate applies for the job listing, an entry is made that the candidate has applied for the job in the database. This is how the details of thousands of candidates applying for the given job listing is stored in the database. At the recruiter's homepage, when the recruiter wants to view the ranklist of all the candidates, they click on 'generate ranklist' option for the given job description. In this case, the TXT file of every candidate resume is compared with the given job description TXT file for calculating the overall resume score. The formula for the same has been mentioned in the design part of the paper. The resume score is out of 100. A certain weightage is given to each component. Each component is also scored based on the requirements given in the job description. For example, a person holding a master's degree will be given a higher score in the education component as compared to a person holding a bachelor's degree in the same field. A person having more number of relevant skills will be assigned with a higher score in the skills component. A person having more number of years of experience will be preferred and hence will be given a higher score. A person with a resume having more cosine similarity with the job description overall will always receive a higher score in that component. On the recruiter's homepage, the ranklist of all the candidates who have applied for the job will be displayed on the basis of the overall resume score based on the factors as given in design. The fields visible to the job recruiter will be the rank, candidate's name, email, phone, and score.

5. CONCLUSION

The prototype developed for this system is primitive and has a lot of scope for betterment in the future. Higher accuracy can be aimed for the NLP model for the future prototypes. A better database option can be used for storing humungous amounts of data of the job applicants as well as the job recruiters. Many job recruiters' accounts can be assimilated with the existing prototype for the flexibility of use for various types of job recruiters. This will fulfill the aim of developing this model for

reducing the workload of the job recruiters in selecting the right employee for their respective company. The most important detail is that there were issues while calculating the number of years of relevant experience for any prospective job candidate. During the calculation of the number of years from the raw unstructured data of the candidate's resume, the number of years of education of the candidate is also getting added along with the number of years of work and internship experience. Hence, manual input had to be provided for the job applicant for the evaluation of the number of years of experience in the overall resume score.

REFERENCES

- [1] Amin, S., Jayakar, N., Sunny, S., Babu, P., M, K. and Gurjar, A. (2019). Web Application for Screening Resume. IEEE Explore (ScopusIndexed).
- [2] Gewirtz, H. (2019). How to Save Jobs by David Gewirtz - docshare.tips.
- [3] Koli, V., Khan, A., Pawaskar, R. and Solaskar, S. (2016). XML Parser for Email Resume System. International Research Journal of Engineering and Technology (IRJET), 3(4), pp.2641-2644.
- [4] Zeck, J., Pain, M., Titano, J., Badgeley, M., Schefflein, J., Su, A., Costa, A., Bederson, J., Lehar, J. and Oermann, E. (2018). Natural Language-based Machine Learning Models for the Annotation of Clinical Radiology Reports | Radiology. [online] Pubs.rsna.org. Available at: <https://pubs.rsna.org/doi/10.1148/radiol.2018171093> [Accessed 28 Jul. 2019].
- [5] En.wikipedia.org. (2018). Use case diagram. [online] Available at: https://en.wikipedia.org/wiki/Use_case_diagram [Accessed 28 Jul. 2019].
- [6] A. Zaroor, M. Maree, and M. Sabha, "A Hybrid Approach to Conceptual Classification and Ranking of Resumes and Their Corresponding Job Posts," In: Czarnowski I., Howlett R., Jain L. (eds) Intelligent Decision Technologies 2017. IDT 2017. Smart Innovation, Systems and Technologies, vol 72. Springer, Cham.