Automated Resume Screener using Natural Language Processing(NLP)

Tumula Mani Harsha
Deparment of CSE
SRM University-AP
Amaravati, Andhrapradesh
India
tumula_mani@srmap.ediu.in

Gangaraju Sai Moukthika
Deparment of CSE
SRM University-AP
Amaravati, Andhrapradesh
India
gangaraju_sai@srmap.edu.in

Dudipalli Siva Sai
Deparment of CSE
SRM University-AP
Amaravati, Andhrapradesh
India
dudipalli_siva@srmap.edu.in

Mannuru Naga Rajeswari Pravallika
Deparment of CSE
SRM University-AP
Amaravati, Andhrapradesh
India
mannuru_naga@srmap.edu.in

Satish Anamalamudi
Deparment of CSE
SRM University-AP
Amaravati, Andhrapradesh
India
satish.a@srmap.edu.in

MuraliKrishna Enduri
Department of CSE
SRM University-AP
Amaravati, Andhrapradesh
India
muralikrishna.e@srmap.edu.in

2022 6th International Conference on Trends in Electronics and Informatics (ICOEI) | 978-1-6654-8328-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICOEI53556.2022.9777194

Abstract—**Resume Screening is the process of evaluating the resume of the job seekers based on a specific requirement. It is used to identify the candidate eligibility for a job by matching all the requirements needed for the offered role with their resume information such as education qualification, skill sets, technical stuff etc. Resume Screening is a crucial stage in candidate's selection for a job role, it is the stage where the decision making is done whether to move the candidate to the next level of hiring process or not. Traditionally, this process is performed manually, but companies often receive thousands of resumes for job applications. In order to reduce the human involvement and errors, many new ways were introduced in this process. This paper discusses about one such process which is very efficient in performing Resume screening. It includes Natural Language Processing (NLP), an automated Machine Learning Algorithm for screening the resumes. This paper explains the end to end working of a python application which efficiently screens the resumes of the candidates based on the organization's requirement.**

Keywords— NLP, Resume Screening, Hiring Process, Skill Set.

I. INTRODUCTION

Effective screening of "Resume" plays a pivotal role in current scenarios due to increased volume of the received applications[1,2]. This screening process requires more domain knowledge to understand the requirement and eligibility for the job role. A large number of job roles are existing currently for which different technologies are required. In these situations, huge number of candidates are applying for the jobs in which the eligible candidates should be selected for which the basic process of Resume Screening should be done[3-6]. Resume Screening is mainly used for the separation of right candidate for the required profile. So, the screening process should be efficient, for which "Resume Screener" can be used. This "Resume Screener" is designed and built by using the Machine Learning approach commonly known as "Natural Language Processing". Figure.1 explains about the job hiring process which basically works with five different steps. Subsequently, Figure.2 explains about the the steps involved in sceening the resumes during the process of job hiring. "Resume Screener" starts working by scanning all the resumes given to it and provides an output regarding the skill sets of the candidates by which the eligible candidates for the requirement can be chosen easily. "Resume Screener" is basically an automated resume recommendation system, which is based on the Machine Learning techniques[7,8]. This model takes the input in Word/PDF format and reflects the output in the PDF, Image and CSV formats. The output will be displayed in the form of graph which includes points for each skill which are required for the job profile[9-12]. According to the output the eligible candidates for the required job can be selected easily. "Resume Screener" is very efficient and less time consuming. It is also practically possible in the current situations[13-16].

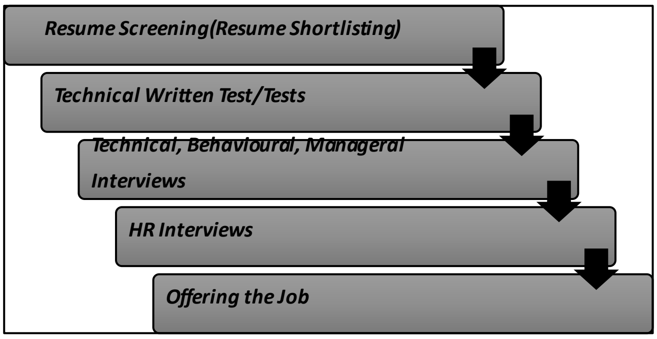

Resume Screening(Resume Shortlisting)
Technical Written Test/Tests
Technical, Behavioural, Manageral Interviews
HR Interviews
Offering the Job

Figure.1. Basic Job Hiring Process

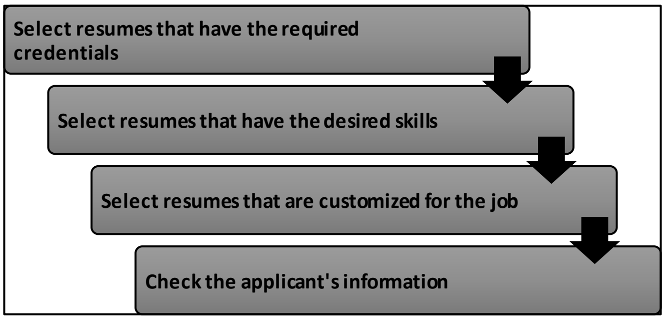

Select resumes that have the required credentials
Select resumes that have the desired skills
Select resumes that are customized for the job
Check the applicant's information

Figure.2. Basic Process of the Resume Screening

II. CHALLENGES IN RESUME SCREENING

a.Time Consuming-Higher level of difficulty if the no. of resumes is high

b.Quality of Hiring-The quality of hiring gets lower when there is bulk of resumes.

c.Hiring Biases-The recruiters can be biased towards some candidates.

d.Recruiter's Experience-If the recruiter is not experienced about all the skills required for the job then the screening process may not be efficient.

e.Recruiters Search-Recruiters may stop the search if the suitable candidate is found and do not go through the entire stack.

f.Unnecessary allocation of recruiters-Separate recruiters should be placed for the resume screening process which results in waste of time and resources[17].

III. EXISTING SOLUTIONS AND PROBLEMS

a. Manual Screening

Manual Screening of resumes is nothing but some of the employees of the company which is going to provide the job will involve in the resume screening process i.e. manually they check each and every application of the candidates resumes one by one till they find their requirement. It can be done based on the skills they want or work experience of the candidates or some other categories which are suitable for the job profile.

Problems

Time Consuming

All the resumes should be referred manually which takes so much of time.

Unnecessary allocation of the resources

Recruiters can involve in any other developments instead of spending so much of time into this resume screening.

Inefficient

Recruiters don't go through all the resumes once they find their requirement.

Biased

The recruiters may be biased towards some of the candidates as per their convenience [18].

b. Resume Screening using Artificial Intelligence

Artificial intelligence, along with text mining and processing algorithms is used to develop these types of Resume Screening Software. These programs usually look for specific keywords to sort resumes and rank them to determine the job applications that should be further reviewed by recruiters[19].

The basic working process

The resume should be in the PDF format then the resume can be opened, read and extract the text. Later the extra text will be removed and the keywords like requirements will be termed area wise. The process gets continued by calculating the scores per each area and gets sorted for final scores creation. Finally based on the scores a pie chart will be displayed as output by which the recruiters can select the required and eligible candidates for the offered job role[20].

Problems

Time Complexity

Scanning each resume at once and calculating the skills takes much time.

Not user friendly

Only the programmer can use the application efficiently.

Tightly coupled blocks

Every code block is dependent on each other to bring any change the whole code should be disturbed which may break the flow [21].

C. Resume Screening using Machine Learning Classifiers

This was built using Machine Learning algorithms and classifiers to screen the resumes[22].

The basic working process

The resumes should be in the CSV format and then the screening process starts with the junk words (unwanted/repeated words) removal from the resumes. Later the remaining words get screened and points for the skills get awarded. And the skills points will be sorted accordingly[23-24]. Finally based on the skill points there will be a graph displayed as the result by which the eligible candidates for the job role can be selected.

Problems

Format Issues

Only CSV format resumes will be considered which is not possible in all situations as the main formats for resumes will be Word or PDF.

Very Complex approach

Idea of dealing with these sorts of approaches will be complex and cannot be understood by everyone[22].

Tightly coupled blocks

Every code block is dependent on each other, to bring any change the whole code should be disturbed which may break the flow[23].

Loss of data

By using some approaches like "genism" (library used for filtering the given data) there might be loss of some important data which might result in the efficiency of the engine.

d.Resume Screening using Deep Learning

Built using the Deep Learning approach and it works by learning from its previous data.

The basic working process

Category: Type of Job Resume fits for.

Resume: Resume of candidates

By using these category and resumes provided, the screening begins. It intakes each resume and screens it and generates output to that particular resume by using natural language processing by resulting in a pie chart. A different graph will be provided per each resume. It will be getting efficient per each run.

Problems

No Bulk resume screening

Only screens one resume per each run and generates output for the single resume.

Time Taking

Higher the resume number, higher the time of screening.

IV. PROPOSED SOLUTION

The proposed solution is designed using Natural Language Processing (a Machine Learning approach) and the Application is named as "Resume Screener". This model takes the input in Word/PDF format and reflects the output in the form of PDF, Image and CSV formats. The goal of the project is to screen the resumes without any manual intervention and provide the efficient result. To achieve this goal, a python project is designed as a web application which can be triggered by APIs.

a. Architectural Overview:

The architecture of this project consists of five blocks as shown in Figure.3.

1.User Interface: From where the requests are made to the application.

2.Application: The part which handles the requests from user interface and performs business logic.

3.Folder part: The places where the resumes are loaded and where the output is stored.

4.Output: The result of the application is stored in different forms based on the screening results.

5. Dataset: A CSV file which holds the skill set.

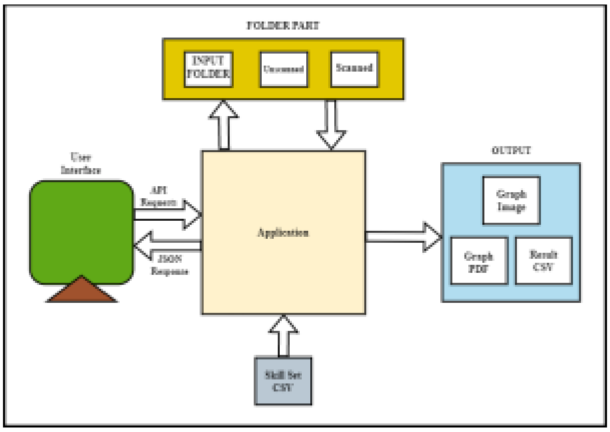


Figure 3: Architechtural overview

b.Functional Overview

The whole flow is divided into four parts.

•The User interface consists of two buttons along with some other info, in which one of the buttons act as format transformer which transforms files (in any other formats) to pdfs. The second button is the key to start the real process which handles the request to screen the pdfs. As soon as this button was hit, the flow starts and the application screens the resumes.

•The folder part of the application consists of four folders in which one is the input folder where resumes are to be loaded, the other is the destination folder in which the resumes transfer after the screening is done, the other one is the unscanned folder in which the resumes which are not scanned properly due to the format issues are stored and the last one is the output folder where outputs are stored. This folder approach makes it easy for the user to distinguish the screened and unscreened resumes.

•The third part of the application is code part which handles the requests from user and performs the process. The application server should be up and running in order to handle the requests from the user.

•The fourth part is output part, it is also a folder in which the output of the application is stored in three formats, PNG, PDF and CSV. Based on the requirement of the user any of the format can be used. Upon each run, these files will be overridden and replaced by the new outputs.

•The fifth part is the Dataset part. It is a CSV file which consists of various skills under various roles. This is used by the application to screen and calculate the frequencies of the resumes. The output graph consists of different frequency points for different skills which will be calculated by comparing the skills of the candidates mentioned in the resume and required skills for the job profile mentioned in the data set. This data set can be updated with different and required skills according to the requirement of the job profile and this can be easy to find the correct fit for the required job.

NLP:

Natural language processing(see Figure.4) is the key approach used in this web application. It is a subfield of computer sciences and artificial intelligence concerned with interactions between the machines and high level language. In simple words it can be stated as how to program applications to process and analyze large amounts of natural language data.

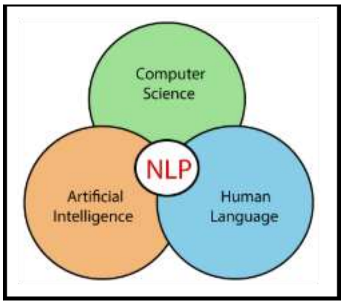


Figure 4: Natural Language Processing

Here in this case we used NLP to scan the text in the resumes by converting the pdf files into plain text files. To do this we have used “en_core_web_sm”. This is used to scan the plain text and to compare this with the skills in skill dataset. Without using NLP, the process might be too complex to find the frequency of skills in each resume.

MVC:

Model View Controller,as shown in Figure.5, is the design pattern which is usually followed by the programmers to make the code readable. According to this approach, the code is divided into layers instead of a single block.

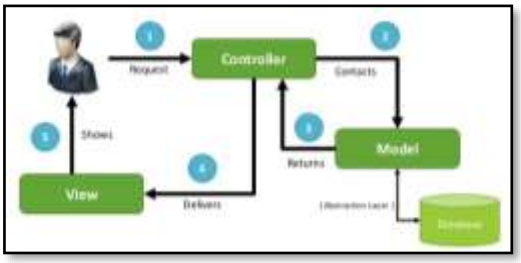
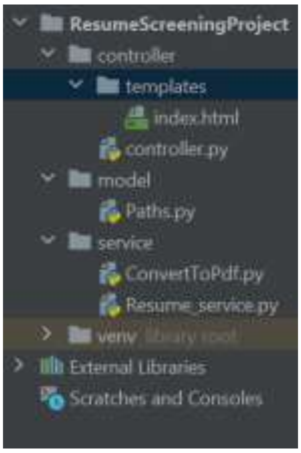


Figure 5: Model View Controller

The main advantage of this process is, each layer is loosely coupled to other layer. If any change is needed inside the code, it can be done in a particular layer without disturbing other layer. In other case, if we write the application as a whole single block of code, any change in the code may break the entire flow.

V. Project Structure

The project structure of the proposed engine is designed as Model, View and Controller model that is shown in Figure.6.



```
ResumeScreeningProject
  controller
    templates
      index.html
    controller.py
  model
    Paths.py
  service
    ConvertToPdf.py
    Resume_service.py
  venv  library root
External Libraries
Scratches and Consoles
```

Figure 6: Project Structure

•Model package consists of the various paths which makes the engine local to any systemin which it is running.
•Service package consists of business logic which converts the input files into pdfs and screens the files based on the input dataset
•Controller package is the one which consists of controller class and a sub package known as templates. The controller class is the one which handles the api requests. The ‘templates’ is the package which responds on the api call and displays the appropriate web pages[22].

VI. Code Walkthrough

The application is basically written in python language and various other techniques are used to make the application programmer friendly and efficient. MVC design pattern is followed in the application, such that the code is divided into different parts based on their functionality.

The total code can be described in four layers,

1. Controller Layer: It is the layer which consists of endpoint description and it is the one which directly communicates with the User Interface. The requests from user interface are directly directed to this layer.

The requests from user are handled in the form of endpoints. Here in this case two endpoints are designed. “/check” to check and convert the files into pdf format and “/screen” to perform the screening operation. The first endpoint provides the successful message after the execution. The second endpoint provides JSON data of the resumes directly on the screen after the execution.

2. Templates/View Layer: This is the layer in which frontend code of the application is written. In a simple way this layer can be stated as the combination of web pages. This layer or package lies inside the controller package.

Here in this layer, mainly two buttons are designed and when each button is clicked, the appropriate endpoint request is sent to the controller.

3. Model Layer: This is the layer which usually consists of the entities or identifiers of the application. In this case, the paths of the folders which are handled by the application is rested here. This makes the programmers to setup the application in different systems easily by changing the paths without disturbing other parts of the code.

4. Service Layer: This is the layer which consists of business logic of the application. In this case this package consists of two service classes. One of them holds the logic to convert the files (doc/docx) into pdfs. The other class holds the logic to screen the resumes based on the skill dataset provided.

Upon "/check" request, the "ConvertToPdf.py" class is triggered and checks the folder in which the resumes are loaded. Then it counts the files with .doc and .docx extensions and converts the files to pdfs. Upon "/screen" request, the "Resume_service.py" class will start the screening of the pdf files in the foler. After screening an internal dataset was build based on the frequencies of the skills provided in the skill dataset. Finally the internal dataset will be plotted as a graph and stored in the output folder along with the csv file. Upon each api call success, a response is displayed on the screen. For '/check' api, the response string "SUCCESSFUL" is displayed and for '/screen' api, the JSON data which is the result of the screening will be rendered upon the successful screening of resumes.

VII. WORKING STRUCTURE

The resumes which should be screened should be placed in the input folder. The resumes can be in the word/pdf format. When the code is compiled and gets run successfully the screening process begins in the web. Initially the input folder consists of the resumes which are to be screened. Later after screening the input folder will be empty as the resumes will be moved to destination. The total working consists of two parts.

1. Coversion of resume files into pdf format

This flow will be triggered by hitting the URL "*/data". Three major formats like PDF, DOC, DOCX are accepted by this application(see Figure.7). But before screening all the docs and docx files are converted into pdfs for smooth flow. Upon successful execution of this flow a message "SUCCESSFUL" will be displayed on the screen.

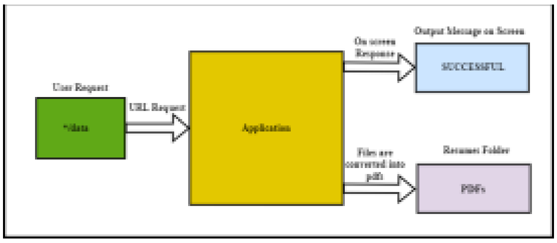


Figure 7: Coversion to pdfs

2. Screening of Resumes

This is the main part of the application. This flow will be triggered by hitting the URL "*/check". The application starts scanning the resumes in the input folder and generates an internal dataset which further calculated and plotted as a bar graph against their name. This part of the application is responsible for movement of files from input folder to destination folder. After Screening the output is stored inside a folder named RESULT. Upon successful execution of this flow the result JSON data is rendered and displayed on the screen(see Figure.8).

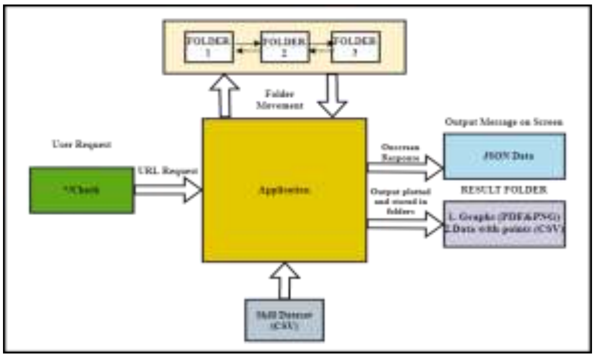


Figure 8: Screening flow

Output: After the screening is performed based on the skillset, the output is stored in various forms like PNG, PDF and CSV in the output folder(Figure.9 and Figure.10).

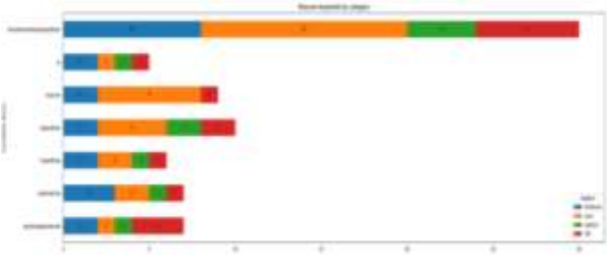
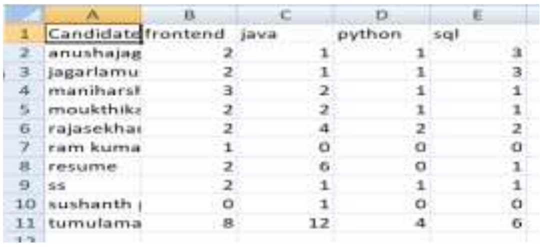


Figure 9: Output Graph (PDF/PNG)



	A	B	C	D	E
1	Candidate	frontend	java	python	sql
2	anushajag	2	1	1	3
3	jagarlamu	2	1	1	3
4	maniharsl	3	2	1	1
5	moukthika	2	2	1	1
6	rajasekhai	2	4	2	2
7	ram kuma	1	0	0	0
8	resume	2	6	0	1
9	ss	2	1	1	1
10	sushanth	0	1	0	0
11	tumulama	8	12	4	6
12					

Figure 10: Output CSV

VIII. ADVANTAGES OF PROPOSED SOLUTION

- Large number of resumes can be screened at a time.
- Attained good time efficiency.
- Multiple formats like .pdf, .doc and .docx are supported by this engine.
- The engine is designed in such a way that the input files(docx/doc/pdf) are converted into pdf's and then screened. This process makes the engine accept multiple formats.
- The NLP is used in such a way that total pdf is converted into plain text and then screened.
- No filtering/deleting of words is involved, such that there will be no extra time utilised to filer each resume.
- The data set provided to the engine is designed in CSV format such that the addition of skills will be easy to the user.
- The blocks of the code are designed in a loosely coupled way, such that the flow will not be disturbed when a new change is added.

- MVC design pattern is used to make the code easily understandable.
- The output is stored in various formats like .png, .pdf and .csv for the convenience.
- The engine is designed in such a way that the files will automatically move to another folder after processing.
- Some resumes which are not properly screened are moved in to a separate folder such that the user can take them in other way.
- Simple User Interface is designed to make it less complex for the user.

IX. CONCLUSION

Resume Screening is one of the most important steps in the recruiting process. Many ways are there to perform this process of screening resumes. Technology has made this Resume Screening process easy for the recruiters. Even a large pool of applications can be screened easily and efficiently with the help of technology. One of the best ways discussed in this paper is the “Resume Screener”. It is very efficient and one of the best ways to screen a large pool of resumes at a time. “Resume Screener” is one of the best way to be used in the resume screening process as it results in the representation which can be very helpful in finding the best fit for different requirements. Finally, “Resume Screener” is a very useful application made using efficient technologies like python and NLP, which results in saving time and resources in the recruiting process for any job role.

REFERENCES

[1] Amit Singh, Catherine Rose, Karthik Visweswariah, Enara Vijil, Nandakishore Kambhatla, “PROSPECT: A system for screening candidates for recruitment”, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010

[2] Kajal Kumari, “Resume Screening with Natural Language Processing in Python”, June 23, 2021.

[3] R. Posevkin and I. Bessmertny, "Translation of natural language queries to structured data sources," 2015 9th International Conference on Application of Information and Communication Technologies (AICT), 2015, pp. 57-59, doi: 10.1109/ICAICT.2015.7338516.

[4] H. Isahara, "Resource-based Natural Language Processing," 2007 International Conference on Natural Language Processing and Knowledge Engineering, 2007, pp. 11-12, doi: 10.1109/NLPKE.2007.4368002.

[5] C. Tapsai, "Information Processing and Retrieval from CSV File by Natural Language," 2018 IEEE 3rd International Conference on Communication and Information Systems (ICCIS), 2018, pp. 212-216, doi: 10.1109/ICOMIS.2018.8644947.

[6] S. Weigelt and W. F. Tichy, "Poster: ProNat: An Agent-Based System Design for Programming in Spoken Natural Language," 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, 2015, pp. 819-820, doi: 10.1109/ICSE.2015.264.

[7] Pradeep KumarRoy, Sarabjeet SinghChowdhary, RockyBhatia, “A Machine Learning approach for automation of Resume Recommendation system”,Procedia Computer Science 167 (2020) 2318–2327.

[8] A. Das and R. C. Balabantaray, "MyNLIDB: A Natural Language Interface to Database," 2019 International Conference on Information Technology (ICIT), 2019, pp. 234-238, doi: 10.1109/ICIT48102.2019.00048.

[9] S. Himani, M. H. Kumar, M. K. Enduri, S. S. Begum, G. Rageswari and S. Anamalamudi, "An Empirical Study on Impact of News Articles," 2021 13th International Conference on Computational Intelligence and Communication Networks (CICN), 2021, pp. 39-43, doi: 10.1109/CICN51697.2021.9574670.

[10] R. Gupta, S. Sameer, H. Muppavarapu, M. K. Enduri and S. Anamalamudi, "Sentiment Analysis on Zomato Reviews," 2021 13th International Conference on Computational Intelligence and Communication Networks (CICN), 2021, pp. 34-38, doi: 10.1109/CICN51697.2021.9574641.

[11] P. Gupta, A. Goswami, S. Koul and K. Sartape, "IQS-intelligent querying system using natural language processing," 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA), 2017, pp. 410-413, doi: 10.1109/ICECA.2017.8212846.

[12] A. Forestiero and G. Papuzzo, "Natural language processing approach for distributed health data management," 2020 28th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2020, pp. 360-363, doi: 10.1109/PDP50117.2020.00061.

[13] Y. Zhang, B. Bogard and C. Zhang, "Development of Natural Language Processing Algorithm for Dental Charting," 2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI), 2020, pp. 403-404, doi: 10.1109/IRI49571.2020.00066.

[14] K. Sintoris and K. Vergidis, "Extracting Business Process Models Using Natural Language Processing (NLP) Techniques," 2017 IEEE 19th Conference on Business Informatics (CBI), 2017, pp. 135-139, doi: 10.1109/CBI.2017.41.

[15] W. Maass, "How visual salience influences natural language descriptions," IEE Colloquium on Grounding Representations: Integration of Sensory Information in Natural Language Processing, Artificial Intelligence and Neural Networks, 1995, pp. 3/1-3/3, doi: 10.1049/ic:19950663.

[16] M. Ahirrao, Y. Joshi, A. Gandhe, S. Kotgire and R. G. Deshmukh, "Phrase Composing Tool using Natural Language Processing," 2021 International Conference on Intelligent Technologies (CONIT), 2021, pp. 1-4, doi: 10.1109/CONIT51480.2021.9498546.

[17] Roberto Salazar, "Resume Screening with Python - Analyzing Candidates Resumes for Jobs Openings", May 3, 2020

[18] FreshWorks, “Resume screening - How to stop swimming in a sea of resumes?”, 2016.

[19] S. Srivastava and S. Lamba, "Hindi-CNL Coder - A Desktop Application for Learning Programming using Native Controlled Natural Language," 2020 IEEE 20th International Conference on Advanced Learning Technologies (ICALT), 2020, pp. 228-229, doi: 10.1109/ICALT49669.2020.00074.

[20] Vamsi Naidu, P., Bommu, V.M.R., Pallapothu, V.D., Janapamula, S.R.N., Kommuri, N.L. (2022). Resume Screening Using Machine Learning. In: Chen, J.IZ., Tavares, J.M.R.S., Iliyasu, A.M., Du, KL. (eds) Second International Conference on Image Processing and Capsule Networks. ICIPCN 2021. Lecture Notes in Networks and Systems, vol 300. Springer, Cham.

[21] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg. Adaptive name matching in information integration. IEEE Intelligent Systems, 18(5):16–23, 2003.

[22] A. Z. Broder. Identifying and filtering near-duplicate documents. In COM ’00: Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching, pages 1–10, London, UK, 2000. Springer-Verlag.

[23] Dharod, Grusha, Harshal Dedhia, Jaineel Shah, Smita Sankhe, and Sneh Chitalia. "Trumer: Rumor Detection on Twitter Using Machine Learning and NLPTechniques." In Innovative Data Communication Technologies and Application, pp. 461-473. Springer, Singapore, 2021.

[24] Dhawale, Apurva D., Sonali B. Kulkarni, and Vaishali Kumbhakarna. "Survey of Progressive Era of Text Summarization for Indian and Foreign Languages Using Natural Language Processing." In International Conference on Innovative Data Communication Technologies and Application, pp. 654-662. Springer, Cham, 2019.