International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 11 Issue: 09 | Sep 2024 www.irjet.net p-ISSN: 2395-0072

Automated Resume Shorlisting

Sanjay [1]

Tech Mahindra

Abstract - This paper presents an automated solution for the candidate selection process in online recruitment using the NLP. In an era where traditional hiring practices prove less effective, our approach addresses the challenges posed by the growing of unstructured resumes. By employing TF-IDF Vectorizer, Truncated SVD, and Cosine Similarity, it accurately evaluates resumes' relevance to job descriptions. Integration of NLTK and PDFMiner ensures precise text processing, handling various resume formats. The system's user-friendly interface, developed with Flask, simplifies interaction, allowing seamless uploading and analysis of resumes. Real-world evaluations demonstrate its efficiency, significantly reducing manual screening efforts. This approach stands as a comprehensive and practical method, showcasing the power of NLP in optimizing the recruitment process for organizations.

Key Words:— TF-IDF Vectorizer, Truncated SVD, and Cosine Similarity NLTK, pdf miner, flask

1.INTRODUCTION

The paper titled "Automated Resume Shortlisting Using NLP: A Comprehensive Approach with TF-IDF Vectorizer, Truncated SVD, Cosine Similarity, NLTK, PDF Miner, and Flask" addresses the challenges faced by HR professionals and recruiters in sorting through a large volume of resumes to identify suitable candidates. The study introduces an innovative solution by integrating Natural Language Processing (NLP) techniques and machine learning algorithms, utilizing advanced tools like TF-IDF Vectorizer, Truncated SVD, Cosine Similarity, NLTK, PDF Miner, and Flask.

The research focuses on automating the resume shortlisting process, making it efficient and accurate. TF-IDF Vectorizer converts resume text into numerical vectors, capturing candidate qualifications. Truncated SVD reduces data dimensionality for faster analysis, while Cosine Similarity matches job requirements with applicant skills, ensuring precise shortlisting. NLTK handles language processing tasks, and PDFMiner extracts text from PDF resumes, broadening document format compatibility. Flask, a Python web framework, creates an interactive interface for user-friendly recruitment.

By automating resume screening, organizations save time and effort in initial recruitment stages. It guarantees fair evaluation, promoting diversity and inclusivity. The paper emphasizes practical implications, demonstrating technical integration and real-world effectiveness. The subsequent sections delve into methodologies, experiments, and results, showcasing the approach's efficacy. The research contributes to HR technology, enabling data-driven hiring decisions and fostering efficient, unbiased recruitment processes.

2. LITERATURE SURVEY

The implementation of Natural Language Processing (NLP) techniques in various domains has significantly revolutionized the way data is processed and interpreted. In the realm of online examinations, NLP has enabled the assessment of descriptive answers, moving beyond the constraints of traditional multiple-choice formats [1]. This shift has been facilitated by employing Python and Django for implementation, providing students with a digital, errorcorrecting experience and ensuring instant evaluation. The versatility of this approach is enhanced by its facultycustomizable questions and keyword-based answer evaluation, leading to a dashboard-driven user interface. Additionally, this innovation has paved the way for the detection of academic dishonesty, thereby enhancing the integrity of examinations [1].

In the domain of requirements extraction, heuristic rules have been explored to extract conceptual models from natural language requirements [2]. This involves identifying concepts through nouns and relationships through verbs, utilizing rules such as compound nouns forming concepts and hierarchical relationships indicated by verbs like 'to be'. This comprehensive approach ensures the extraction of relevant models from natural language requirements, contributing to effective requirement analysis and system design.

In the context of automated resume shortlisting, NLP algorithms have been instrumental in parsing resumes and social profiles automatically, transforming unstructured data into a structured format [5]. This innovation addresses the challenge of extracting structured information from diverse resume formats. The ranking process, which includes attributes such as education, experience, and communication skills, is handled efficiently through this automated system. By employing techniques such as cosine similarity and machine learning models like K-Nearest Neighbor and Support Vector Machine, the system achieves accurate and efficient automated ranking for client companies [5] [10].

Furthermore, NLP has been employed in skill analysis, where techniques like Word2Vec and bigrams are utilized to streamline recruitment processes [11]. By integrating NLP algorithms such as SBERT and cosine similarity, resume screening is optimized, swiftly identifying job matches, accommodating diverse languages and unstructured data [12]. Additionally, NLP algorithms have been utilized for structuring text data and topic modeling, contributing to resume scoring based on job descriptions [13].

In the domain of search engine ranking functions, various techniques, including BM25 and LM-DS, have been explored and compared using particle swarm optimization [7]. The study emphasizes the effectiveness of stemming and the superiority of combining stemming and feedback methods for improved ranking function performance. This comparative analysis provides valuable insights into the effectiveness of different ranking functions, paving the way for enhanced information retrieval systems.

Moreover, machine learning methods such as Random Forest, Naive Bayes, and Logistic Regression have been applied for resume classification, utilizing libraries like NLTK, SpaCy, and Scikit-learn [14]. These methods, coupled with matching algorithms such as Cosine Similarity and Levenshtein Distance, contribute to accurate skill matching and resume classification. This integration of machine learning techniques and matching algorithms ensures precise identification of candidates suitable for specific job roles.

In summary, the integration of NLP techniques, machine learning models, and various algorithms has significantly advanced the fields of online examinations, requirements extraction, and automated resume shortlisting. These innovations have not only enhanced the efficiency and accuracy of processes but also opened avenues for further research and improvements in these domains.

3. METHODOLOGY

This section outlines a robust methodology for creating an advanced resume shortlisting system. It combines various techniques to automate candidate selection, ensuring efficiency and accuracy in the process. The proposed methodology is structured systematically, offering a detailed framework for seamless integration and implementation. It encompasses diverse strategies, enhancing the overall effectiveness of resume screening and optimizing the recruitment workflow.

Data Collection and Preprocessing: Our research initiative commenced with the acquisition of a diverse dataset of resumes sourced from prominent job portals, encompassing a wide array of industries, experience levels, and job roles. These resumes, often stored in PDF format, were meticulously processed using the PDFMiner library. Text extraction from PDFs was followed by a rigorous cleansing process, where special characters, formatting discrepancies, and irrelevant information were expunged. This ensured the extracted text was pristine and ready for analysis.

CGPA Filtering: Recognizing the significance of academic achievements, we incorporated an additional layer of qualification. Candidates' Cumulative Grade Point Average (CGPA) was extracted from the resumes. To streamline the shortlisting process, a stringent criterion was applied: only candidates with a CGPA greater than or equal to 8 were considered for further analysis. This step ensured that candidates not only possessed professional experience but also demonstrated exceptional academic performance.

Skill Extraction and Dataset Preparation: To automate the process of resume shortlisting, a comprehensive skills dataset was essential. Leveraging an existing skills repository, we meticulously curated a dictionary containing an extensive list of technical, soft, and domain-specific skills. This dataset served as the foundation upon which the skills mentioned in the resumes were matched, allowing for a nuanced analysis of applicants' qualifications.

Text Vectorization using TF-IDF: Raw textual data, inherently unstructured, underwent a transformational process leveraging the power of TF-IDF vectorization. This technique converted the textual content into numerical vectors, essential for subsequent machine learning analysis. By assigning TF-IDF scores to each term in the corpus, the importance of terms within individual resumes was weighed against their prevalence across the entire dataset.

Dimensionality Reduction using Truncated SVD: The resultant high-dimensional TF-IDF vectors were subjected to the efficiency-enhancing process of Truncated Singular Value Decomposition (SVD). By significantly reducing the dimensionality of the data while retaining its essential information, Truncated SVD facilitated streamlined computations and optimized model performance.

Cosine Similarity Calculation: Cosine similarity, a pivotal metric in our methodology, quantified the likeness between resumes. Computed from the reduced TF-IDF vectors, cosine similarity scores provided a measure of resemblance between applicants' qualifications. Resumes with higher cosine similarity scores indicated a more substantial overlap in skills and experiences, a vital factor in the shortlisting process.

Natural Language Processing (NLP) for Skill Context Enhancement: To further enhance the accuracy of skill matching, advanced Natural Language Processing (NLP) techniques were deployed. Tokenization, stemming, and lemmatization, implemented via the NLTK library, transformed both the skills dataset and resume text. These processes ensured that various forms of a skill term were

recognized uniformly, refining the matching accuracy and minimizing discrepancies.

Development of Flask Web Application: To provide an intuitive and accessible interface for resume shortlisting, a robust web application was developed using the Flask framework. Within this platform, users could seamlessly upload resumes in PDF format. Upon upload, the application processed the resumes using the integrated NLP and machine learning techniques, ensuring a smooth user experience.

Integration of Skills Matching Algorithm: Central to the Flask application was the seamlessly integrated skills matching algorithm. Leveraging the preprocessed skills dataset, TF-IDF vectors, truncated SVD components, and cosine similarity calculations, the algorithm performed a comprehensive analysis. When a user uploaded a resume, the algorithm processed the text, calculated cosine similarity scores against the skills dataset, and identified the most pertinent skills mentioned. Candidates meeting the stringent CGPA criterion were ranked based on these similarity scores, presenting the most qualified applicants to the user.

In conclusion, our meticulous methodology encompassed CGPA filtering, data preprocessing, skill extraction, TF-IDF vectorization, dimensionality reduction, cosine similarity calculation, and advanced NLP techniques. These elements synergistically contributed to the development of an automated resume shortlisting system, ensuring the identification of highly qualified candidates for further consideration in the recruitment process.

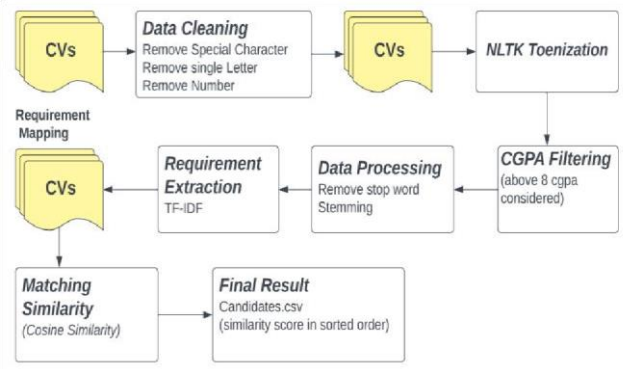


Fig.1.Resume Shortlisting Flowchart

Mathematical Explanation

Vectorization of TF-IDF and Reduction of Dimensionality determined the Term Frequency (TF) for every term in the CV and job description:

TF (i, A) = (Total number of terms in A) / (Number of times term i appears in A)

TF (i, B) = (Total number of terms in B) / (Number of times word i appears in B)

Determine each term's Inverse Document Frequency (IDF): IDF(i) = log (N / n) where n is the number of documents that include the phrase i and N is the total number of documents.

Determine each term's TF-IDF value in the two documents:

TF-IDF (i, A) = TF (i, A) * IDF(i)

TF-IDF (i, B) = TF (i, B) * IDF(i)

Use Principal Component Analysis (PCA) to make the TFIDF vectors less dimensional. Say that you want to reduce the dimensionality to k dimensions.

Compute Cosine Similarity

After PCA, you will get reduced-dimension TF-IDF vectors A' and B'. You may now compute their cosine similarity:

Determine the vectors A' and B' dot product: For all elements i, the Dot Product (A' · B') = Σ (A'[i] * B'[i]). The reduced-dimension TF-IDF values for the ith term in the vectors A' and B' are denoted by the values A'[i] and B'[i].

Determine vector A's magnitude (Euclidean norm):

For all elements i, magnitude A' (|A'|) = √ (Σ (A'[i]^2)

Determine vector B's magnitude (Euclidean norm):

For all elements i, magnitude B' (|B'|) = √ (Σ (B'[i]^2)

Using the dot product and magnitudes, get the cosine similarity (cos(θ)) between vectors A' and B': Similarity between cosines (cos(θ)) = (A' · B') / (|A'| * |B'|) When using PCA for dimensionality reduction, the necessary information in the TF-IDF vectors is preserved while a significant reduction in computational complexity and memory utilization is achieved. You can adjust the parameter k, or the number of dimensions, to suit your particular requirements.

Example Data

Assume the following terms are present in our TF-IDF vectors:

Words: ["develop", "medical", "software", "programming", "experience"]

TF-IDF Vector (A): A = [0.2, 0.3, 0.4, 0.1, 0.2] is the job description.

Restart the TF-IDF Vector (B) by writing B = [0.1, 0.2, 0.3, 0.5, 0.4].

PCA-Assisted Dimensionality Reduction

Suppose we wish to use PCA to decrease the dimensionality to two dimensions:

Reduced-dimension vectors A' and B' are obtained by applying PCA on vectors A and B.

Compute Cosine Similarity

The cosine similarity between the reduced-dimension vectors A' and B' will now be determined:

Determine the vectors A' and B' dot product:

Dot Product (A' · B') = (A'[1] * B'[1]) + (A'[2] * B'[2]) Utilizing the formula Magnitude, A' (|A'|) = √((A'[1] ^2) + (A'[2] ^2), find the magnitude of vector A'.

Determine the vector B's magnitude using the formula:

Magnitude B' (|B'|) = √((B'[1] ^2) + (B'[2] ^2))

Determine the similarity of cosine:

Similarity between cosines (cos(θ)) = (A' · B') / (|A'| * |B'|) Let's do a step-by-step calculation of these values: (0.2 * 0.1) + (0.3 * 0.2) = 0.02 + 0.06 = 0.08 is the dot product of (A' · B').

Magnitude A' (|A'|): -√ ((0.2^2) + (0.3^2)) = -√ (0.04 + 0.09)

= -0.13 ≈ 0.36

Magnitude B' (|B'|): -√ ((0.1^2) + (0.2^2)) = -√ (0.01 + 0.04)

= -0.05 ≈ 0.22

(A' · B') / (|A'| * |B'|) = (0.08) / (0.36 * 0.22) ≈ 0.97 is the cosine similarity (cos(θ))

Thus, there is a significant degree of similarity between the job description and the resume, as indicated by the cosine similarity of roughly 0.97 between the reduced-dimension TF-IDF vectors A' and B'. Usually, use libraries or other tools to carry out these computations in practice.

4. RESULTS

The deployment of Natural Language Processing techniques enhances the precision of skill matching, while the development of a user-friendly Flask web application streamlines the interaction between recruiters and the automated shortlisting system.

Figure 2 illustrates the overall interface of the Automated Resume Shortlisting Web App. This visual representation showcases the user interface and the primary components of the application. It likely provides an overview of the software layout, including the different sections and features available to the users. This figure serves as a baseline reference for readers to understand the interface design of the automated resume shortlisting system.

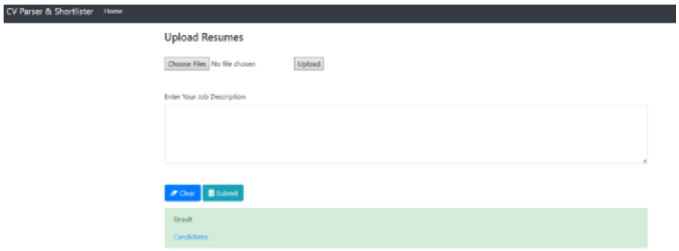


Fig.2.Automated Resume Shortlisting Web App

Fig.3 depicts the resumes uploaded for the parsing and shortlisting purpose. It can be clearly seen that here 2 resumes are uploaded with .pdf extension. The allowed document type is varied such as. docs,.pdf etc.

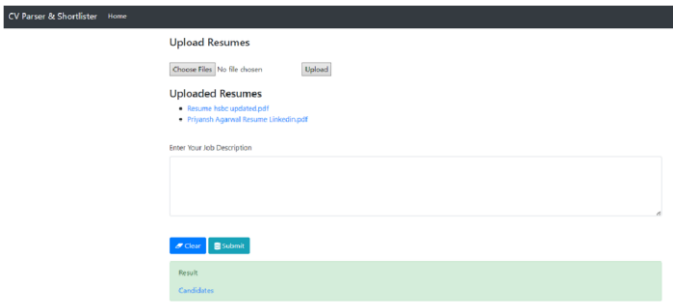


Fig.3. Automated Resume Shortlisting Web App

With uploaded resumes

Fig.4 shows that user have added the job description for the shortlisting of the candidates with the highest similarity score It demonstrates the seamless integration of job requirements, showcasing the system's capability to compare uploaded resumes with specific job criteria.

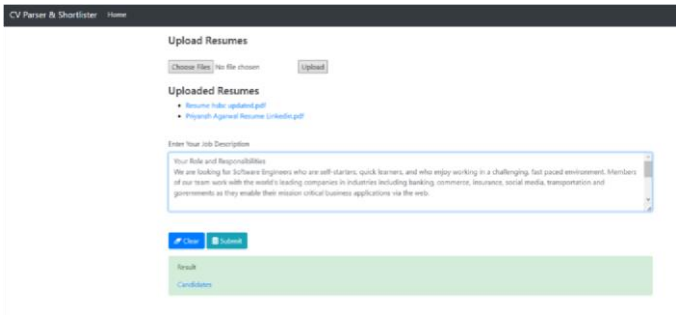
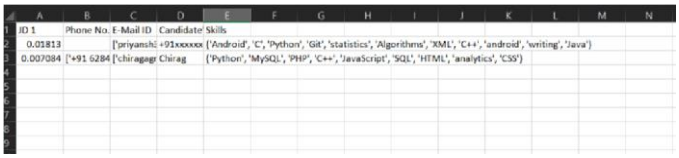


Fig.4 Automated Resume Shortlisting Web App With uploaded resumes and Job Description

Fig.5 shows the excel sheet in which candidates are listed out in sorted way in sequence of highest similarity score



ID 1	Phone No.	E-Mail ID	Candidate Skills
0.01813		['priyanshi	+91xxxxxx {'Android', 'C', 'Python', 'Git', 'statistics', 'Algorithms', 'XML', 'C++', 'android', 'writing', 'Java'}
0.007084	['+91 6284	['chiragagr	Chirag {'Python', 'MySQL', 'PHP', 'C++', 'JavaScript', 'SQL', 'HTML', 'analytics', 'CSS'}

Fig.5.Candidate's similarity score in sorted order

5. LIMITATIONS AND FUTURE SCOPE

While our automated resume shortlisting system demonstrates significant efficiency, it has limitations. The current system may face challenges with highly complex job descriptions and nuanced skill requirements. Additionally, The CGPA criterion may not be universally applicable, limiting its use in industries where it's not a relevant metric.

Future research can focus on refining the skills dataset, incorporating deep learning techniques for more nuanced skill matching, and exploring natural language understanding for context-aware evaluation. Integration with online professional platforms and continuous learning algorithms could enhance the system's adaptability, ensuring it remains effective in dynamically changing job markets. Furthermore, user experience enhancements and real-time analytics could be explored for more comprehensive recruitment solutions.

6. CONCLUSION

In this study, we have presented a sophisticated approach to automated resume shortlisting using Natural Language Processing (NLP) techniques, we have covered step by step implementations for resume shortlisting. By integrating Cumulative Grade Point Average (CGPA) filtering, we ensured that candidates not only possessed professional experience but also demonstrated exceptional academic excellence.

Our methodology, employing TF-IDF vectorization, Truncated Singular Value Decomposition (SVD), and cosine similarity calculation, facilitated accurate and efficient matching of candidates' skills against job requirements.

This research not only enhances the efficiency of resume screening but also reduces the time and effort invested in manual shortlisting. As the demand for skilled professionals continues to rise, our automated resume shortlisting system stands as a valuable tool in streamlining the hiring process, ensuring that organizations can swiftly identify and engage with the most qualified candidates

7. REFERENCES

[1] Jayaram Nori Broadcom Inc,

USA.DOI:-

https://www.doi.org/10.56726/IRJMETS53503

[2] P. Prasad and A. Rao, "A survey on various challenges and issues in implementing AI for enterprise monitoring," Journal of Network and Computer Applications, vol. 116, pp. 42-55, 2018, doi: 10.1016/j.jnca.2018.05.005.

[3] Y. Dang, Q. Lin, and P. Huang, "AIOps: real-world challenges and research innovations," in 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), 2019, pp. 4-5, doi: 10.1109/ICSE-Companion.2019.00023.

[4] D. Xu et al., "Unsupervised anomaly detection via variational auto-encoder for seasonal KPIs in web applications," in Proceedings of the 2018 World Wide Web Conference, 2018, pp. 187-196, doi: 10.1145/3178876.3185996.

[5] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," ACM Computing Surveys (CSUR), vol. 41, no. 3, pp. 1-58, 2009, doi: 10.1145/1541880.1541882.

[6] M. Chen et al., "Big data: A survey," Mobile Networks and Applications, vol. 19, no. 2, pp. 171-209, 2014, doi: 10.1007/s11036-013-0489-0.

[7] Y. Li et al., "Deep learning for anomaly detection in cloud native systems," in 2020 IEEE International Conference on Cloud Engineering (IC2E), 2020, pp. 106-116, doi: 10.1109/IC2E48712.2020.00022.

[8] F. Salfner, M. Lenk, and M. Malek, "A survey of online failure prediction methods," ACM Computing Surveys (CSUR), vol. 42, no. 3, pp. 1-42, 2010, doi: 10.1145/1670679.1670680.

[9] F. Jiang et al., "Artificial intelligence in healthcare: Past, present and future," Stroke and Vascular Neurology, vol. 5, no. 2, 2020, doi: 10.1136/svn-2020-000443.

[10] F. Salfner, M. Lenk, and M. Malek, "A survey of online failure prediction methods," ACM Computing Surveys (CSUR), vol. 42, no. 3, pp. 1-42, 2010, doi: 10.1145/1670679.1670680.

[11] X. Liu et al., "PANDA: Facilitating usable AI development," arXiv preprint arXiv:2003.04070, 2020.

[12] G. A. Susto, A. Beghi, and C. De Luca, "A predictive maintenance system for epitaxy processes based on filtering and prediction techniques," IEEE Transactions on Semiconductor Manufacturing, vol. 25, no. 4, pp. 638-649, 2012, doi: 10.1109/TSM.2012.2209131.

[13] X. Liu et al., "PANDA: Facilitating usable AI development," arXiv preprint arXiv:2003.04070, 2020.

[14] E. Cortez et al., "Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms," in Proceedings of the 26th Symposium on Operating Systems Principles (SOSP), 2017, pp. 153-167, doi: 10.1145/3132747.3132772.

[15] Z. Yin et al., "An empirical study on configuration errors in commercial and open source systems," in Proceedings of the 26th Symposium on Operating Systems Principles (SOSP), 2017, pp. 159-176, doi: 10.1145/3132747.3132773.

[16] D. Wang et al., "Failure prediction using machine learning in a virtualised HPC system and application," Cluster Computing, vol. 20, no. 1, pp. 103-115, 2017, doi: 10.1007/s10586-016-0668-4.

[17] J. Gao, "Machine learning applications for data center optimization," Google White Paper, 2014. [Online]. Available: https://research.google/pubs/pub42542/

[18] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in 2010 IEEE Symposium on Security and Privacy, 2010, pp. 305-316, doi: 10.1109/SP.2010.25.

[19] R. Boutaba et al., "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," Journal of Internet Services and Applications, vol. 9, no. 1, pp. 1-99, 2018, doi: 10.1186/s13174-018-0087-