



RESUME SCREENING AND JOB MATCHING SYSTEM USING NATURAL LANGUAGE PROCESSING

S.Selsiya¹, Dr. K. Merrilance M.Sc., M.Phil., Ph.D²

Department of Computer Application, Sarah Tucker College, Tirunelveli

ABSTRACT :

Resume screening is an important task in the recruitment process, but manually reviewing a large number of resumes is time-consuming and may lead to inconsistent evaluation. To address this problem, this paper proposes a Resume Screening and Job Matching System that automates the initial screening of candidate resumes based on job requirements. The proposed system is implemented as a web application using Python and Flask. Resume documents in PDF format are processed using Natural Language Processing techniques to extract and preprocess relevant textual information. Term Frequency-Inverse Document Frequency (TF-IDF) is used to represent the resume and job description as numerical vectors, while Cosine Similarity is used to measure the textual similarity between them. In addition, skill matching is performed to identify relevant and missing skills. A hybrid matching score is generated by combining textual similarity and skill matching, allowing candidates to be ranked according to their relevance to a particular job. The system also provides candidate application management and status tracking features. The proposed approach aims to reduce manual screening effort, improve consistency in initial candidate evaluation, and provide a simple and efficient solution for automated resume screening and job matching.

1. INTRODUCTION

Recruitment is an important organizational process that involves identifying and selecting suitable candidates for specific job vacancies. In the recruitment process, organizations receive a large number of resumes containing information about educational qualifications, technical skills, work experience, certifications, projects, and other competencies. Recruiters need to examine these resumes and compare the information with the requirements specified in job descriptions. When this process is performed manually, it can require considerable time and effort, particularly when the number of applications is large. Manual screening may also result in inconsistent comparison because different resumes can present similar skills and qualifications using different words or formats.

Resumes are generally unstructured documents, whereas job descriptions contain structured requirements related to a particular position. This difference makes automated resume-job matching a text processing problem. Simple keyword-based approaches can identify specific terms present in a resume, but they may not adequately represent the overall textual relationship between a resume and a job description. Therefore, Natural Language Processing (NLP) techniques can be used to preprocess and analyse textual information [3], [5], in order to improve the systematic comparison of these documents.

Several text representation and similarity techniques can be applied to document-matching problems. Term Frequency-Inverse Document Frequency (TF-IDF) represents documents as numerical feature vectors [1], [2] by assigning importance to terms based on their frequency and relevance. Cosine Similarity can then be used to measure the similarity between the vector representations of a resume and a job description [2], [8]. These techniques provide an interpretable and computationally practical approach for establishing a baseline for automated resume screening. In addition to document-level similarity, explicit skill matching can provide more detailed information about whether the candidate possesses the skills required for a particular position.

This study proposes a research-oriented Resume Screening and Job Matching System that combines TF-IDF, Cosine Similarity, and skill matching. The system accepts job requirements and candidate resumes in PDF format. The text from the uploaded resume is extracted and subjected to preprocessing to remove irrelevant information and prepare the content for analysis. The processed resume and job description are represented using TF-IDF, and Cosine Similarity is calculated to determine their textual relevance. A separate skill-matching process identifies the skills present in the candidate resume and compares them with the required skills specified for the job. The textual similarity and skill-matching results are then combined to generate a hybrid matching score.

The generated score provides a basis for comparing and ranking candidates according to their relevance to a particular job requirement. The system also identifies matched and missing skills, making the screening result more interpretable than relying only on a single similarity value. In addition to the research component, the web-based system supports job management, candidate applications, candidate selection or rejection, and application status tracking. These features integrate the research-based matching process with the initial recruitment workflow.

The main significance of this study is the combination of general textual similarity and explicit skill coverage in a single interpretable matching approach. The current implementation can be considered a practical baseline for resume-job matching rather than a complete benchmark evaluation. Future research can extend this work by using larger labelled resume-job datasets, systematic evaluation metrics, improved skill normalization,

synonym mapping, semantic embeddings, and transformer-based models such as BERT [6]. Such extensions can help investigate whether advanced semantic approaches provide improvements over traditional TF-IDF and similarity-based methods. Thus, the proposed system establishes a foundation for further research and development in automated resume screening and job matching.

2. RELATED WORK

Several studies have investigated the use of Natural Language Processing (NLP) and text-mining techniques for automating resume screening and job matching. Earlier approaches mainly focused on extracting keywords, skills, educational qualifications, and experience from resumes and comparing them with job requirements. Keyword-based methods are relatively simple to implement and can identify specific skills present in a candidate profile. However, these approaches may provide limited information when the same skill or qualification is expressed using different terms.

Text similarity techniques have also been explored for comparing resumes with job descriptions. Term Frequency-Inverse Document Frequency (TF-IDF) is commonly used to represent textual documents by assigning importance to terms according to their frequency and relevance. Cosine Similarity can then be applied to measure the similarity between the numerical representations of resumes and job descriptions. These methods are computationally simple and provide an interpretable approach for document matching.

Recent studies have explored semantic approaches to overcome some limitations of traditional keyword and frequency-based methods. Word embeddings can represent words based on their contextual relationships [10], [11], allowing systems to identify similarities between related terms. Transformer-based models such as BERT provide more advanced contextual representations [6], and can capture semantic relationships in text. However, these approaches generally involve greater computational requirements and more complex model implementation.

Based on these approaches, the present study combines document-level similarity using TF-IDF and Cosine Similarity with explicit skill matching. This combination is intended to provide both an overall textual comparison and a skill-level comparison between resumes and job requirements. The approach provides an interpretable baseline for automated resume screening while also establishing a foundation for future research using semantic and transformer-based techniques.

3. SYSTEM WORKFLOW

The company first creates a job vacancy by entering the job title, job description, and required skills. A candidate views the available job and uploads a PDF resume. The system extracts resume text using pypdf and preprocesses both the resume text and job description. TF-IDF converts the processed text into numerical vectors. Cosine Similarity measures document-level similarity, while a separate skill-matching stage identifies matching and missing skills. These two components are combined into a hybrid score, and candidates can be ranked according to their final match scores.

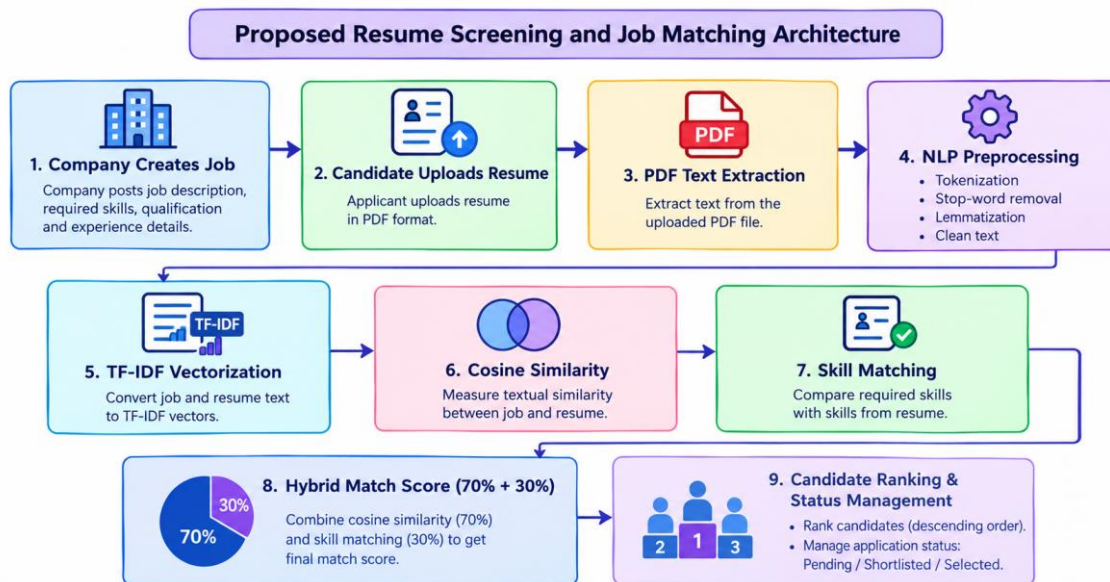


Figure 1. Architecture of the Proposed Resume Screening and Job Matching System

ARCHITECTURE EXPLANATION

- 1. Company Creates Job**
Company enters the job description, required skills, qualifications, and experience details.
- 2. Candidate Uploads Resume**
Candidate uploads the resume in PDF format.

3. **PDF Text Extraction**

The system extracts the resume content from the uploaded PDF file.

4. **NLP Preprocessing**

The extracted text is cleaned using tokenization, stop-word removal, lemmatization, and other preprocessing techniques.

5. **TF-IDF Vectorization**

The job description and resume text are converted into TF-IDF vectors to identify important terms.

6. **Cosine Similarity**

The system calculates the similarity between the job description and the resume using cosine similarity.

7. **Skill Matching**

The required skills in the job description are compared with the skills identified from the resume.

8. **Hybrid Match Score (70% + 30%)**

The final score is calculated by combining **70% Cosine Similarity** and **30% Skill Matching**.

9. **Candidate Ranking & Status Management**

Candidates are ranked in descending order based on their match score, and application statuses such as **Pending, Shortlisted, and Selected** are managed.

4. MODULE DESCRIPTION

Module 1: Company Job Management Module

This module allows the company to log in and create job vacancies by specifying job title, job description, and required skills. It also provides access to candidate applications submitted for each job.

Module 2: Candidate Job Viewing and Application Module

Candidates can view available jobs and submit application details along with a resume in PDF format. The system stores the application information and associates it with the selected job.

Module 3: Resume PDF Text Extraction Module

The uploaded PDF resume is processed using the PdfReader functionality from the pypdf library. Text is extracted from the resume and prepared for further Natural Language Processing.

Module 4: NLP Preprocessing Module

The extracted resume text and job description are cleaned before similarity calculation. Stop-word removal and normalization reduce irrelevant textual information so that important terms have greater influence on the comparison.

Module 5: TF-IDF Feature Extraction Module

Term Frequency-Inverse Document Frequency transforms the processed resume and job description into numerical feature vectors. TF-IDF gives relatively higher importance to informative terms and lower importance to very common terms.

Module 6: Cosine Similarity Module

Cosine Similarity measures the angle between the TF-IDF vectors of the resume and job description. A higher cosine similarity value represents stronger textual similarity between the two documents.

Module 7: Skill Matching Module

The system compares the required skills of the job with skills identified in the candidate resume. It generates lists of matched skills and missing skills to make the screening result more interpretable.

Module 8: Hybrid Match Score and Ranking Module

The current implementation combines 70% TF-IDF similarity and 30% skill matching to generate the final match score. Candidates can be ordered according to this score to support initial screening.

Module 9: Candidate Selection and Status Module

The company can mark an application as Pending, Selected, or Rejected. Candidates can later check their application status using their registered email and phone number.

Module 10: Web Interface and Database Module

Flask provides the web interface and routing, while SQLite stores company, job, and candidate information. The application integrates the screening workflow with job posting, candidate management, and status tracking.

5. MATCH SCORE

The proposed Resume Screening and Job Matching System generated a **match score of 35.14%** for the selected candidate and job requirement. The match score represents the overall relevance between the candidate's resume and the specified job description. It is calculated using the hybrid matching approach implemented in the system, which combines textual similarity and skill matching. The textual similarity component uses TF-IDF representation and Cosine Similarity to compare the content of the resume with the job description. The skill-matching component compares the required skills with the skills identified in the candidate's resume.

A score of 35.14% indicates that the candidate has a certain level of relevance to the given job requirements, but several required skills are not present or are not identified in the resume. The system therefore categorizes this result as a **Moderate Match**. The matched skills include Communication, CSS, Flask, Git, HTML, Machine Learning, NLP, and Python, while missing skills include Natural Language Processing, NumPy, Pandas, Problem Solving, Scikit-learn, SQL, and Teamwork.

This result demonstrates that the system provides more than a single numerical score by also presenting skill-level information. The score should be considered an observed application result rather than an accuracy or classification performance metric.

5.1 MATCH CATEGORY – MODERATE MATCH

The **Match Category “Moderate Match”** indicates that the candidate's resume has a reasonable level of similarity with the requirements mentioned in the job description, but it does not completely satisfy all the required qualifications and skills. This category is generated based on the overall match score obtained by comparing the candidate's resume with the job requirements.

In the proposed Resume Screening and Job Matching System, the candidate's resume is processed using Natural Language Processing techniques. The system extracts important information such as education, experience, technical skills, and keywords from the uploaded resume. It then compares this information with the job description using **TF-IDF and Cosine Similarity**. In addition, the system checks whether the required skills are present in the candidate's resume.

When the calculated score falls within the moderate matching range, the system classifies the candidate as a **Moderate Match**. This means that the candidate may possess some relevant skills and qualifications, but certain important requirements are missing or not identified in the resume. Therefore, the candidate may require further review by the recruiter before making a final selection.

This category helps recruiters quickly understand the suitability of a candidate and reduces the time required for manual resume comparison.

5.2 MATCHED SKILLS

The **Matched Skills** represent the skills identified in the candidate's resume that are also relevant to the requirements specified in the job description. In the obtained result, the system identified the following matched skills: **Communication, CSS, Flask, Git, HTML, Machine Learning, NLP, and Python**. These skills indicate that the candidate possesses several technical and general skills related to the job requirements.

The system identifies these skills during the resume processing stage after extracting text from the uploaded PDF resume. The extracted resume content is preprocessed using Natural Language Processing techniques to remove unnecessary words and normalize the text. The processed resume is then compared with the combined job requirements. Along with TF-IDF and Cosine Similarity, the system performs explicit skill matching to identify relevant skills present in the candidate's resume.

For example, **HTML and CSS** indicate web development-related skills, while **Python, Flask, Git, Machine Learning, and NLP** represent programming, framework, development, and data-processing-related capabilities. **Communication** represents a general skill relevant to professional roles. The matched skills provide recruiters with a clear explanation of why a candidate received a particular matching score. Thus, skill-level matching improves the interpretability of the automated screening process and supports further recruiter review.

5.3 MISSING SKILLS

The **Missing Skills** represent the skills mentioned in the job requirements that were not identified in the candidate's resume during the matching process. In the obtained result, the system identified **Natural Language Processing, NumPy, Pandas, Problem Solving, Scikit-learn, SQL, and Teamwork** as missing skills. These missing skills indicate that the candidate's resume does not contain these required skills in a form that was recognized by the system.

The system identifies missing skills by comparing the required skills specified in the job description with the skills extracted or detected from the candidate's resume. This comparison is performed along with the text-based matching process using **TF-IDF and Cosine Similarity** and explicit skill matching. If a required skill is not detected in the resume, it is listed under the missing skills category.

For example, **NumPy, Pandas, and Scikit-learn** are technical skills, while **SQL** represents a database-related skill. **Problem Solving and Teamwork**

are general professional skills. **Natural Language Processing** is also a relevant technical requirement; however, the system may identify it separately from the abbreviation **NLP**, depending on the exact text representation used for matching. Therefore, the missing-skills list helps recruiters understand which requirements need further verification and provides additional explanation for the candidate's overall match result.

Result Item	Observed Value
Match Score	35.14%
Match Category	Moderate Match
Matched Skills	Communication, CSS, Flask, Git, HTML, Machine Learning, NLP, Python
Missing Skills	Natural Language Processing, NumPy, Pandas, Problem Solving, Scikit-learn, SQL, Teamwork

In one application test, the system generated a match score of 35.14% and classified the candidate as a Moderate Match. Eight skills were identified as matched and seven were identified as missing for that particular resume-job combination. This example illustrates how the system exposes both the overall score and the underlying skill-level comparison.

6. DISCUSSION

The experimental output shows that the proposed system can automate the initial stage of resume screening by comparing candidate resumes with job descriptions and providing understandable feedback to recruiters. The system uses TF-IDF and Cosine Similarity to measure the textual similarity between the resume and job requirements. In addition, explicit skill matching identifies the specific skills that are present or missing in the candidate's resume. By combining these two approaches, the hybrid method provides both general text-based relevance and job-specific skill information. This makes the screening result more interpretable for recruiters.

However, the current implementation should be considered a practical baseline rather than a complete benchmark evaluation. The observed result from the application represents the outcome of a single test case and should not be considered as accuracy, precision, recall, or F1-score. To calculate reliable classification or prediction performance metrics, a larger labelled dataset and a clearly defined ground-truth evaluation procedure are required.

The discussion also identifies a limitation in exact or phrase-based skill matching. For example, "NLP" and "Natural Language Processing" may be treated as different skills if normalization rules are not applied. This limitation can be addressed in future work through synonym mapping, skill ontologies, word embeddings [10], [11], and transformer-based semantic models [6], [12].

7. CONCLUSION

The proposed Resume Screening and Job Matching System provides an automated approach for evaluating candidate resumes against job requirements. The system combines Natural Language Processing, TF-IDF, Cosine Similarity, and skill matching to generate an interpretable hybrid match score. The Flask-based web application also supports company login, job posting, candidate resume submission, candidate ranking, selection or rejection, and application status tracking.

By automating the initial screening process, the system can reduce manual effort and provide a consistent preliminary comparison of candidates. The current implementation establishes a practical research-oriented baseline for resume-job matching. Future work can focus on larger benchmark datasets, improved skill normalization, semantic similarity, BERT-based models, and systematic comparison with other machine learning approaches.

8. FUTURE ENHANCEMENT

Future enhancements include the use of larger and standardized resume datasets, semantic skill normalization, multilingual resume processing, advanced embeddings, and transformer-based models such as BERT [6]. The system can also be extended with recruiter analytics, automated recommendations, more detailed candidate profiles, and systematic evaluation using labelled resume-job pairs. These extensions can support more accurate semantic matching and stronger experimental validation.

9. REFERENCES

- [1] Ramos, J. (2003). Using TF-IDF to Determine Word Relevance in Document Queries.
- [2] Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.
- [3] Jurafsky, D., & Martin, J. H. *Speech and Language Processing*. Pearson.
- [4] Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- [5] Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media.
- [6] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of NAACL-HLT*.
- [7] Salton, G., & McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill.

-
- [8] Salton, G., Wong, A., & Yang, C. S. (1975). A Vector Space Model for Automatic Indexing. *Communications of the ACM*, 18(11), 613–620.
- [9] Manning, C. D., Manning, C. D., & Raghavan, P. (2009). *An Introduction to Information Retrieval*. Cambridge University Press.
- [10] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.
- [11] Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. *Proceedings of EMNLP*, 1532–1543.
- [12] Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *Proceedings of EMNLP-IJCNLP*, 3982–3992.
- [13] Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems*, 30.
- [14] Lewis, M., Liu, Y., Goyal, N., et al. (2020). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *Proceedings of ACL*, 7871–7880.
- [15] McCallum, A. K., & Nigam, K. (1998). A Comparison of Event Models for Naive Bayes Text Classification. *AAAI-98 Workshop on Learning for Text Categorization*, 41–48.
- [16] Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning with Many Relevant Features. *Proceedings of ECML*, 137–142.
- [17] Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., & McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit [17]. *Proceedings of ACL System Demonstrations*, 55–60.
- [18] Loper, E., & Bird, S. (2002). NLTK: The Natural Language Toolkit. *Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, 63–70.
- [19] Harris, Z. S. (1954). Distributional Structure. *Word*, 10(2–3), 146–162.
- [20] Robertson, S., & Zaragoza, H. (2009). The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends in Information Retrieval*, 3(4), 333–389.