



e-ISSN: 2319-8753 | p-ISSN: 2347-6710

# IJIRSET

International Journal of Innovative Research in  
**SCIENCE | ENGINEERING | TECHNOLOGY**



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 13, Issue 11, November 2024

**ISSN** INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA

**Impact Factor: 8.524**

 9940 572 462

 6381 907 438

 [ijirset@gmail.com](mailto:ijirset@gmail.com)

 [www.ijirset.com](http://www.ijirset.com)

# Cumulative Score based Resume Score Generation and Ranking

**Prof. Reeta Koshy, Shaan Agarwal, Yash Burad, Shivam Kamble**

Assistant Professor, Department of Computer Engineering, Sardar Patel Institute of Technology, Andheri,  
Mumbai, India

U.G. Student, Department of Computer Engineering, Sardar Patel Institute of Technology, Andheri, Mumbai, India

U.G. Student, Department of Computer Engineering, Sardar Patel Institute of Technology, Andheri, Mumbai, India

U.G. Student, Department of Computer Engineering, Sardar Patel Institute of Technology, Andheri, Mumbai, India

**ABSTRACT:** In today's competitive job market, resume screening is a critical step in recruitment, but traditional manual review methods are time-consuming and subjective. This paper presents a cumulative score-based resume scoring and ranking system designed to automate the evaluation process. The system uses natural language processing (NLP) and machine learning techniques to accurately assess key candidate qualifications, including education, experience, and skills. Our model incorporates an abbreviations dictionary and utilizes FuzzyWuzzy for text similarity scoring, ensuring flexibility in matching degrees and job titles. Additionally, we calculate experience-based scores based on role durations, applying rewards or penalties based on match accuracy with job requirements. Our approach leverages technologies like the Google Gemini API for data extraction, LangChain for structured document parsing, FAISS for efficient similarity search. This paper demonstrates how an automated scoring model can streamline resume screening, reduce human bias, and support fair candidate selection.

**KEYWORDS:** ATS, LangChain, FAISS vector store, NLP, Google Gemini, PyPDF, FuzzyWuzzy.

## I. INTRODUCTION

Resume screening plays a central role in recruitment, but it often involves manual review, leading to challenges such as inconsistency, subjectivity, and time inefficiency. As candidate pools grow, hiring teams face increasing pressure to process resumes quickly while maintaining accuracy in selecting the best-fit candidates [1]. Recent advancements in NLP and artificial intelligence (AI) have opened avenues for automating resume assessments, allowing for a more standardized and efficient evaluation process.

This research focuses on developing a cumulative score-based model that automates resume scoring by analysing education, experience, and skill match with job descriptions. The model integrates several advanced tools and methods. A key feature of our approach is an abbreviations dictionary, enhancing degree and title matching even when specific terminology varies. We also introduce a scoring system for experience, calculating job duration in months and applying bonuses for excess experience or penalties for shortfalls. This resume scoring tools reduce human error, improve efficiency, and enable recruiters to focus on higher-value tasks like interviews and decision-making

The purpose of this paper is to demonstrate how an automated, data-driven approach to resume screening can improve recruitment efficiency and fairness. Through this system, recruiters can achieve a more objective and consistent selection process, reducing biases and ensuring that qualified candidates are accurately identified.

## II. LITERATURE SURVEY

In recent years, traditional resume screening processes have been increasingly challenged by the sheer volume of applications, which makes manual screening both time-consuming and prone to human error. To address these issues, several automated resume screening systems have been proposed, integrating advanced machine learning and natural language processing (NLP) techniques to streamline the recruitment process.

The [2] research paper presents a framework for resume parsing that leverages Named Entity Recognition (NER) and machine learning techniques, specifically utilizing the Boolean Naive Bayes algorithm for effective categorization of resume sections and the challenges posed by diverse layouts, including tables and infographics. By converting unstructured resume data into structured outputs, the study shows candidate profile matching and ranking, ultimately improving the efficiency of the e-recruitment process.

The research paper [3] presents a solution using Information Extraction (IE) techniques, aims to streamline the recruitment process by transforming unstructured resume data into a structured format. Information like personal details, education, and skills are extracted, and can be used by HR professionals in identifying suitable candidates. The authors emphasize that effective communication is crucial in recruitment, suggesting that features like real-time feedback for users and integration with communication tools could enhance collaboration among hiring teams.

The research paper [4] explores the critical role of Automatic Text Summarization (ATS) in managing the vast amounts of information available online. It distinguishes between two primary summarization techniques: extractive summarization, which selects key sentences from the original text, and abstractive summarization, which generates new sentences that encapsulate the main ideas. The paper highlights the capabilities of GPT-3 in performing both types of summarization, emphasizing its potential for fine-tuning to enhance performance on specific tasks.

The research paper [5] presents an automated Resume Classifier that utilizes Natural Language Processing (NLP) techniques. It employs a Phrase Matcher to evaluate resumes against specific recruiter criteria, calculating scores based on extracted information. The classifier is benchmarked against traditional models like Support Vector Machines and Naive Bayes. The [6] research paper proposes an automated resume screening system that utilizes a 2Q-Learning framework to extract and evaluate skills from resumes. However, a significant gap in this approach is its limited focus on skills, as it does not adequately consider the candidate's experience, which is a crucial factor in assessing overall suitability for job roles.

### III. METHODOLOGY

In this research, we propose a comprehensive resume scoring and ranking system that automatically extracts key information from resumes and evaluates them based on specific criteria against job descriptions. This system employs Google's Gemini API for parsing resumes, LangChain for document processing, and FAISS for vector storage. The system is designed to assess resumes by calculating an Automated Tracking System (ATS) score based on degree, skills, and experience matching criteria, then ranks candidates according to their scores.

#### 3.1 Data Extraction from Resumes Using Google Gemini API:

We begin by utilizing Google Gemini API to extract data from resumes in PDF format, with a predefined prompt designed to retrieve candidate information in a structured JSON format. The API is integrated using LangChain and operates as follows:

**3.1.1. Resume PDF Content Processing:** Resumes uploaded by candidates for particular job vacancy are read using PyPDF2 library, and their text content is extracted page by page. This text is then divided into chunks of 10,000 characters with a 1,000-character overlap using LangChain's RecursiveCharacterTextSplitter to ensure coherent and manageable portions for further processing.

**3.1.2 Vector Store Creation:** Extracted text chunks are embedded using Google Generative AI embeddings. These embeddings are then indexed and stored in a FAISS vector store, enabling efficient similarity searches in resume context.

**3.1.3 Information Extraction:** We designed a prompt template in LangChain to engage the Google Gemini model. We instruct the Gemini API to extract candidate details from the resume in JSON format, by providing the context from the resume, ensuring robust information extraction.

**3.1.4: Output Structure:** After parsing the information from resume, the JSON format includes the section like,

- Personal Information: Name, email, and contact details.
- Education: Includes fields like college name, degree, skills learned, marks, and time period.

- Experience: Captures job roles, companies, technologies used, skills acquired, and duration.
- Projects: Details such as project name, description, tools, technologies, and skills.

This data is stored in JSON format for use in the scoring phase.

### 3.2. ATS Score Calculation:

Once structured resume data is available, an ATS score is calculated based on multiple factors aligned with job description requirements. The scoring process includes degree, skills, and experience matching using fuzzy logic for precise similarity calculations.

**3.2.1 Degree matching:** The degree matching algorithm checks for a match between the candidate's qualifications and the job's required degree. We implemented an abbreviation expansion feature to recognize common educational abbreviations using a custom dictionary.

For example:

"BE" is expanded to "Bachelor of Engineering"

"BTech" is expanded to "Bachelor of Technology"

"CS" is expanded to "Computer Science".

For similarity checking, we use FuzzyWuzzy library to calculate similarity scores between the job required degree and the candidate's degree. We used this library's with below comparison strategy:

- `fuzz.token_set_ratio`: This function tokenizes both strings (splits them into individual words) and calculates a similarity ratio based on the unique tokens. It focuses on matching the words in any order. This is particularly useful for degrees, where words like "Bachelor" or "Engineering" may appear in different sequences.
- `fuzz.partial_ratio`: This computes similarity by focusing on the best matching substring in one string against the entire other string, which is helpful when one string is a subset of the other. For example, if the job description requires "Bachelor of Science in Computer Science," and the candidate has "Bachelor of Science," `partial_ratio` will still yield a reasonable similarity score.

We use the maximum of `token_set_ratio` and `partial_ratio` to get a combined score. If this score meets a 70% similarity threshold, we consider the degrees a match and add 2 points to the `ats_score`.

**3.2.2. Skills Matching:** The skills matching component compares the skills in the candidate's resume with both the required and preferred skills in the job description. Skills are extracted from the education, past roles, and project sections. The scoring methodology includes:

- Exact Matches: Each exact skill match from required core skills adds 2 points, while matches with preferred technologies add 1 point.
- Partial Matches: If a skill is not an exact match but partially resembles a required skill, a smaller score (0.25 points) is awarded.

The FuzzyWuzzy library's `fuzz.ratio()` function is used to calculate the similarity between the required skills and user skills. This is the foundation of fuzzywuzzy. For example, changing "developer" to "developing" would require 3 edits (remove r, add g, replace e with i), yielding a Levenshtein distance of 3. The smaller the distance, the more similar the two strings are.

**3.2.3. Experience Matching:** The experience matching component evaluates how well a candidate's work duration aligns with the job's experience requirements. It calculates the total months of experience by converting each past job's start and end dates into months.

- If the candidate meets or exceeds the required experience, they receive 2 points. Extra months add more points (0.01 points for each extra month). If the candidate falls short of the required experience, 0.01 will be deducted from score per short month.

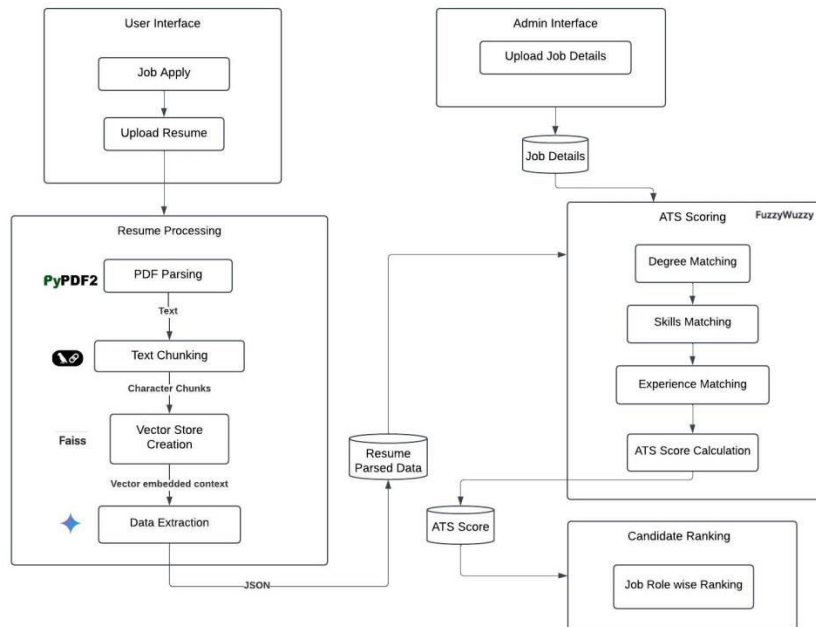


Fig 1: Architecture Diagram

**- Technology Stack:**

Our implementation uses the following technologies:

**Google Gemini API:** This API is used for advanced data extraction and natural language processing (NLP) capabilities, allowing us to efficiently pull relevant information from documents. It supports complex NLP operations which improve the accuracy of data retrieval in resume parsing and scoring tasks.

**LangChain:** LangChain manages the process flow for document parsing by chaining together tasks based on pre-defined prompts. The tool follows prompt-based instructions for a streamlined, automated parsing sequence, ensuring consistent and structured extraction for each resume.

**FAISS (Facebook AI Similarity Search):** FAISS is a powerful library for efficient similarity searching, particularly with vector data. In this system, it stores and quickly retrieves similar resume segments or terms, which aids in rapid, accurate matching of candidate qualifications to job descriptions.

**FuzzyWuzzy:** This library provides a method for measuring string similarity, which is critical for matching degree names, skills, and job titles, even when phrasing varies. FuzzyWuzzy helps ensure that near-matches are scored accurately, even if there are minor textual differences in a candidate's resume.

**PyPDF2:** PyPDF2 handles PDF parsing, allowing text extraction from resumes in PDF format. By reading and processing content directly from PDF files, PyPDF2 ensures that information is captured accurately, regardless of file format, for further analysis.

**Flask:** Flask serves as the backend framework, responsible for rendering pages, communicating with the database, and sending data to the frontend. It handles the server-side logic and ensures seamless interaction between the user interface and backend components.

**MySQL Database:** The MySQL database stores key information, including candidate data, resume files paths, job descriptions, and ATS score at admin side. It ensures that data is efficiently stored and easily accessible for further processing and analysis.

#### IV. EXPERIMENTAL RESULTS

Below are the Results and Screenshot of Application, where Candidate can Apply for the job, After Uploading the resume, he/she is able to see the resume details which are editable and submittable. Finally, ATS score will be stored based on our algorithm and candidates will be ranked.

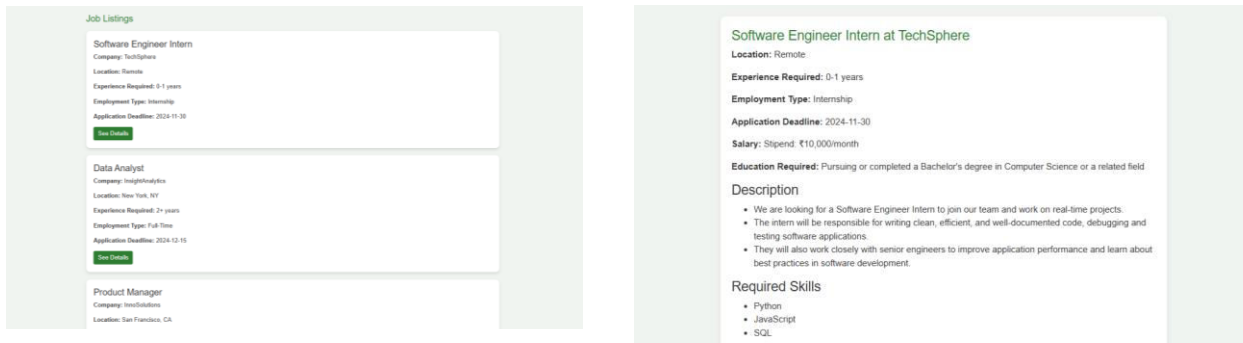


Fig 2.: Candidate Applying for the job

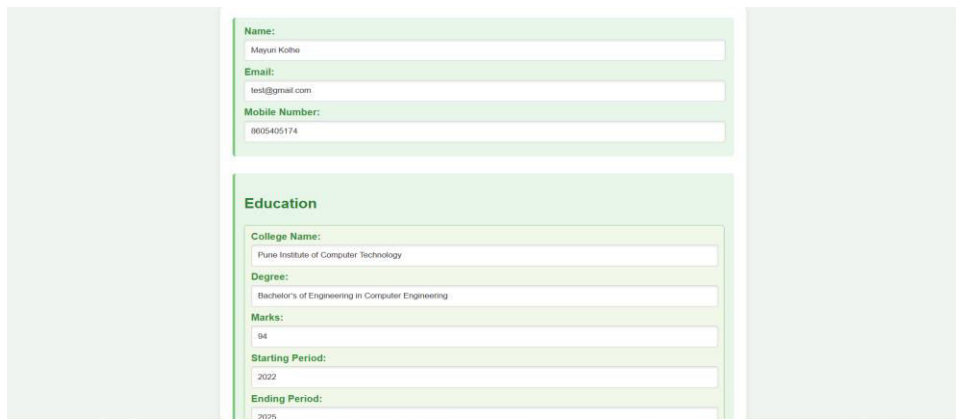


Fig 3: Basic Information and Educational Information is extracted from Resume through parsing and shown in the form for any edits

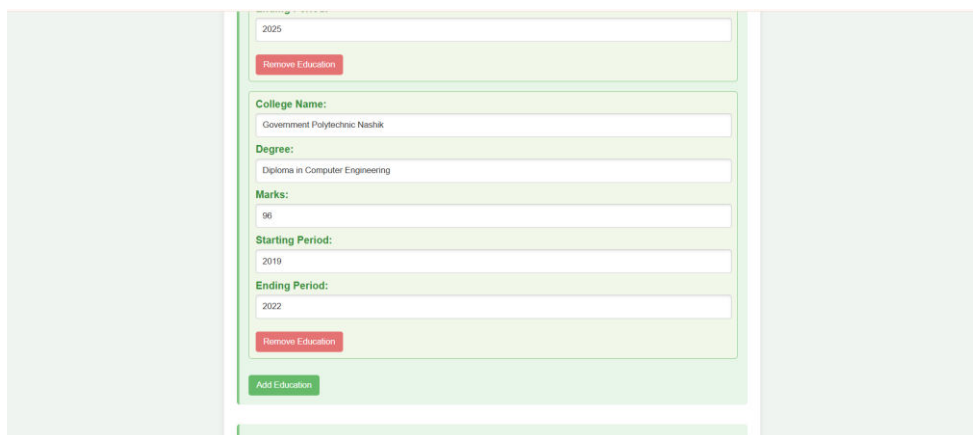


Fig. 4: Candidate can Add Education / Remove Education if missed by Resume Parser while extracting information

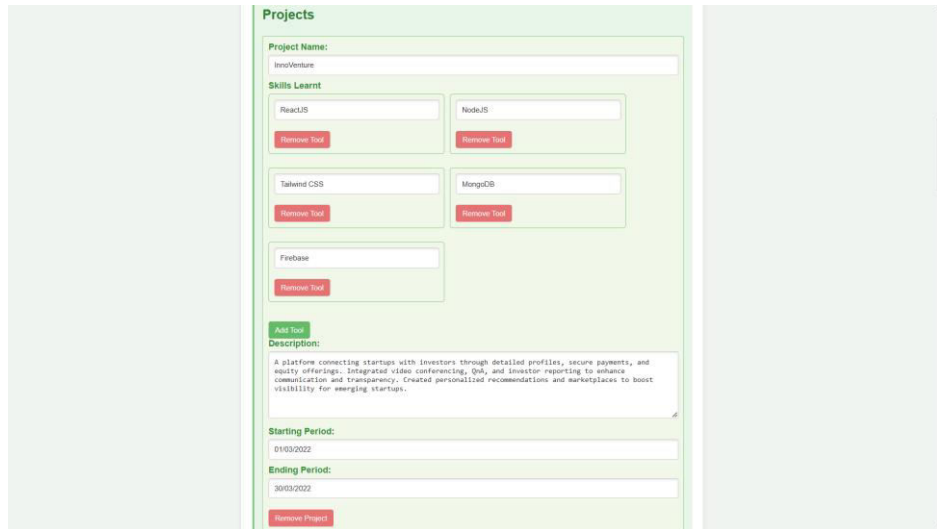


Fig. 5: Extraction and showing Project related information like Skills, Description, Duration. Candidate can add or remove them as well.

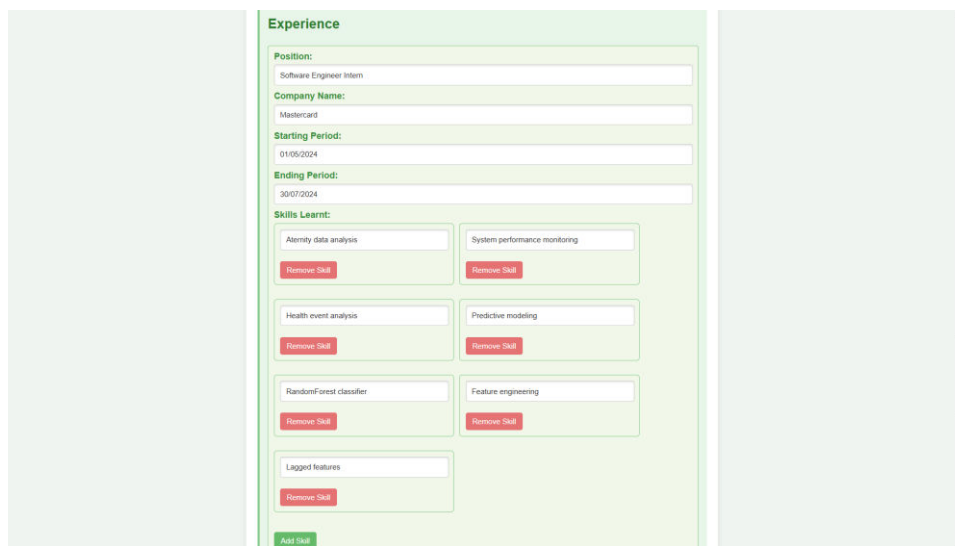


Fig. 6: Extraction and showing past experiences of candidates with addition and removal of skills, addition and removal of particular Role/Experience. Candidate will submit the form if all information is true.

Resume Ranking for Software Engineer Intern Position

Rank	Candidate Name	ATS Score	View Resume
1	Arjun Mehta	26.13	<a href="#">View Resume</a>
2	Mayuri Kolhe	24.73	<a href="#">View Resume</a>
3	Shivam Patil	22.48	<a href="#">View Resume</a>
4	Pooja Patil	16.26	<a href="#">View Resume</a>

Fig. 7: ATS score of candidates and Descending order Ranking for particular job role.

**Data Extraction:** We accurately extracted text from resumes using the PyPDF2 library, ensuring compatibility across different formatting styles. We tested multiple resumes, we successfully got text. After receiving a prompt, Google Gemini processes the input using advanced natural language understanding models to interpret and extract relevant information. It analyses the document's content based on the prompt's instructions, identifies key sections (e.g., personal details, experience, skills), and outputs the extracted data in a structured format, such as JSON.

**ATS score Scoring:** We tested the system extensively against various job descriptions and resumes. It accurately matches skills, even with slight differences in wording, thanks to the FuzzyWuzzy library's capability to recognize near matches. The system also correctly assesses required experience based on the job criteria. The Google Gemini API demonstrates high intelligence in extracting information from the entire resume content and organizing it into a structured JSON format with impressive accuracy, capturing all relevant details for the scoring process.

## V. CONCLUSION

In today's fast-paced recruitment environment, efficiently ranking resumes is essential to identify the best-fit candidates quickly and accurately. Our research presents a cumulative score-based resume scoring and ranking system that optimizes this critical stage of recruitment. By implementing advanced resume parsing, scoring, and ranking methodologies, our system enables hiring managers to streamline candidate selection and make informed decisions. Using Google Gemini API, LangChain, and FAISS vector storage, we developed a robust framework to extract, structure, and manage resume data with high precision. This process generates a JSON output of candidate information, allowing users to make edits if necessary. Given the Gemini API's leading accuracy in natural language processing, the system minimizes the risk of data inaccuracies, further supporting reliable candidate evaluation.

The automated scoring process assesses resumes based on skill, degree, and experience matches with job requirements, offering a data-driven alignment with recruiter expectations. By generating a comprehensive ATS score, our model highlights candidates who best meet the job's qualifications, promoting an objective and consistent recruitment process. This research contributes a valuable tool for recruiters, helping them fast-track the hiring process while maintaining a high standard of candidate quality and fit for organizational goals.

## REFERENCES

1. Indeed Editorial Team. "Resume Screening for Recruiters (Definition and How-To)." Indeed, <https://www.indeed.com/career-advice/resumes-cover-letters/resume-screening>. Accessed [November 5, 2024].
2. Rimmalapudi, R., Rankela, S. S. H., Joseph, B., Duggineni, V. L., & Alapati, J. R. (2024). Resume parsing using named entity recognition and Hugging Face model. *International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)*, 13(3), 2585-2591.A.
3. Bagde, A., Desai, S., Khamankar, S., Sahane, R., Korhale, M., & Pingat, A. (2024). Resume Parsing Web-App Using AI. *International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)*, 13(5).
4. Wankhade, S. B., Ghanekar, R., Das, D., Kamath, V., & More, N. (2023). Text summarization using GPT-3. *International Journal of Innovative Research in Science, Engineering and Technology*, 12(4), 3572-3580.
5. Bachhav, R., Deshmukh, M., Muqeema, M., & Bhaladhare, P. R. (2024). Automated text analysis: Leveraging NLP for categorization. *International Journal of Innovative Research in Science, Engineering and Technology*.
6. Bhoomika, S. P., Likhitha, S., Chandana, H. S., Kavva, S. A., & Bhargavi, K. (2022). 2Q-Learning Scheme for Resume Screening. In *Proceedings of the 4th International Conference for Emerging Technology (INCET)*





INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

 9940 572 462  6381 907 438  [ijirset@gmail.com](mailto:ijirset@gmail.com)



[www.ijirset.com](http://www.ijirset.com)

Scan to save the contact details