International Journal of Science, Strategic Management and Technology
Volume **02** Issue **04** April-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

AI-Driven Resume Skill Extraction and Job Recommendation System using Hybrid Transformer Mamba Model

Bala Amithesh. S[1], Dr. M. Ramnath[2], Dr. M. Kaliappan[3]

[1]UG student, Department of Artificial Intelligence and Data Science, Ramco Institute of Technology, Rajapalayam, Tamil Nadu, India

[2]Assistant Professor (S.G.), Department of Artificial Intelligence and Data Science, Ramco Institute of Technology, Rajapalayam, Tamil Nadu, India

[3]Professor, Department of Artificial Intelligence and Data Science, Ramco Institute of Technology, Rajapalayam, Tamil Nadu, India

[1]balaamithesh07@gmail.com, [2] ramnath25@gmail.com,[3]kalsrajan@yahoo.co.in

https://doi.org/10.55041/ijsmt.v2i4.199
Cite this Article: S, B. A. (2026). AI-Driven Resume Skill Extraction and Job Recommendation System using Hybrid Transformer Mamba Model. International Journal of Science, Strategic Management and Technology, *02*(04). https://doi.org/10.55041/ijsmt.v2i4.199

ABSTRACT

Manual resume screening is a time-consuming and error-prone process, particularly in large-scale recruitment where organizations receive hundreds of applications per job opening. Traditional Applicant Tracking Systems (ATS) primarily rely on rule-based keyword matching, which lacks contextual understanding and often leads to false positives and false negatives during candidate shortlisting. This paper presents an AI-driven Resume Skill Extraction and Job Recommendation System that integrates a custom-trained MAMBA Named Entity Recognition (NER) model with TF-IDvectorization and cosine similarity-based ranking. The proposed framework performs automated resume parsing, contextual skill identification, skill normalization, job-role similarity computation, and recommendation generation. Unlike conventional keyword-based systems, the MAMBA model captures contextual skill entities, improving extraction accuracy and adaptability to varied resume formats. Extracted skills are transformed into numerical vectors using TF-IDF weighting, and cosine similarity is computed between candidate profiles and predefined job role descriptions to determine relevance scores. Experimental evaluation demonstrates an estimated real-world F1-score of approximately 0.88, representing a significant improvement over traditional rule-based ATS systems while maintaining low computational overhead. The system executes efficiently on standard CPU hardware, making it suitable for small and medium-scale enterprise deployment. The proposed architecture offers a scalable, interpretable, and cost-effective solution for intelligent recruitment automation, balancing accuracy and computational efficiency.

Keywords: Resume Screening, Named Entity Recognition (NER), SpaCy, TF-IDF, Cosine Similarity, Recruitment Automation, Natural Language Processing (NLP), Applicant Tracking Systems (ATS)

1. INTRODUCTION

The digital transformation of recruitment platforms has significantly increased the volume of job applications processed per vacancy. Organizations frequently receive hundreds or even thousands of resumes for a single role. Manual screening under such conditions is inefficient, inconsistent, and prone to human bias. Traditional Applicant Tracking Systems (ATS) primarily employ rule-based keyword matching and dictionary lookup mechanisms. These systems fail to interpret contextual relationships between skills and experience. For instance, a candidate experienced in "Deep Learning frameworks such as TensorFlow and PyTorch" may be overlooked if the system searches only for the keyword "Machine Learning Engineer" without contextual mapping.

Advancements in Natural Language Processing (NLP) enable contextual understanding of unstructured text data. Named Entity Recognition (NER) models can extract structured information from resumes by recognizing skill entities within their contextual usage.

This research proposes a lightweight yet robust AI-based framework combining:

- Custom SpaCy NER for contextual skill extraction

- TF-IDF vectorization for numerical representation

- Cosine similarity for job recommendation ranking

The objective is to design a scalable and interpretable system that improves recruitment efficiency without requiring expensive GPU infrastructure.

2. RELATED WORK

Automated resume screening has evolved considerably with the advancement of Natural Language Processing and machine learning technologies. Early recruitment systems relied heavily on rule-based Applicant Tracking Systems that used predefined keyword dictionaries, Boolean search logic, and regular expression matching to filter candidate resumes.

Although computationally efficient, these systems lacked contextual understanding and failed to recognize semantic relationships between related skills, resulting in high false-negative rates and biased filtering outcomes. For example, variations in terminology such as "data analytics" and "data analysis" were often treated as unrelated concepts, thereby reducing system reliability. To overcome these limitations, researchers introduced classical machine learning models including Support Vector Machines, Naïve Bayes classifiers, Random Forest algorithms, and Logistic Regression approaches, which utilized feature engineering techniques such as Bag-of-Words representation, n-gram extraction, and frequency-based scoring. While these methods improved classification accuracy compared to rule-based systems, they still depended heavily on manual feature design and struggled with contextual generalization across domains.

Subsequent research focused on Named Entity Recognition models to extract structured information such as skills, educational qualifications, work experience, and certifications directly from unstructured resume text. Sequence labeling methods including Conditional Random Fields and Bidirectional LSTM architectures significantly enhanced contextual extraction performance by learning word dependencies. More recently, transformer-based architectures such as BERT and its variants have demonstrated superior performance in entity recognition and semantic similarity tasks, achieving high F1-scores across benchmark datasets. However, these models require large annotated datasets, GPU acceleration, high memory consumption, and increased inference time, limiting their practical deployment in resource-constrained environments.

3. LITERATURE REVIEW

Ali et al. (2021) developed a resume classification system using Natural Language Processing (NLP) and machine learning techniques to automate the screening process. Their approach focuses on extracting relevant textual features from resumes and classifying them into predefined categories based on job requirements. The system significantly reduces manual effort and improves efficiency in recruitment by leveraging supervised learning models. It also highlights the importance of preprocessing techniques such as tokenization and feature extraction to enhance classification accuracy, demonstrating that automated systems can outperform traditional manual screening methods in terms of speed and consistency [1].

Honnibal and Montani (2017) introduced spaCy, an advanced NLP framework designed for efficient and scalable language processing. Their work emphasizes the use of neural networks, word embeddings, and incremental parsing for tasks such as Named Entity Recognition (NER), part-of-speech tagging, and dependency parsing. In the context of resume screening, spaCy plays a crucial role in extracting structured information such as skills, education, and experience from unstructured text. The model's speed and accuracy make it highly suitable for real-time applications, forming a strong foundation for building intelligent recruitment systems [2].

Kinge et al. (2021) proposed a resume screening system that integrates machine learning and NLP techniques to match candidate profiles with job descriptions. Their system processes resumes by extracting keywords and relevant features, which are then used to train classification models. The study highlights the effectiveness of automated screening in reducing bias and improving the consistency of candidate evaluation. It also demonstrates how machine learning algorithms can enhance decision-making in recruitment by identifying the most suitable candidates based on data-driven insights [3].

Pedregosa et al. (2011) presented Scikit-learn, a comprehensive machine learning library in Python that provides simple and efficient tools for data mining and analysis. The library includes essential functionalities such as TF-IDF

4. METHODOLOGY

The proposed methodology follows a structured computational pipeline designed to automate resume skill extraction and job recommendation while maintaining efficiency and scalability. The system begins with resume acquisition, supporting both PDF and DOCX formats, followed by text extraction using document parsing libraries to convert unstructured files into processable textual data.

Extracted text undergoes normalization procedures including lowercasing, removal of punctuation and special characters, tokenization, stopword elimination, and lemmatization to ensure consistent linguistic representation. This preprocessing stage reduces noise and prepares the input for contextual entity recognition. A custom-trained SpaCy Named Entity Recognition model is then employed to identify skill entities based on contextual word relationships rather than static dictionary matching. The training dataset consists of annotated resume sentences labeled with a custom SKILL entity tag, and the model is optimized using iterative gradient-based learning.

Unlike rule-based systems, the contextual NER approach recognizes multi-word skills, framework names, programming languages, and domain-specific terminologies even when phrased differently across resumes.

Once skill entities are extracted, they are transformed into numerical representations using the Term Frequency–Inverse Document Frequency weighting scheme, which assigns higher importance to distinctive skills while reducing the impact of frequently occurring generic terms. The TF-IDF representation ensures a compact and interpretable vector space suitable for similarity computation. Cosine similarity is subsequently applied to measure the angular distance between resume vectors and predefined job description vectors, producing a normalized relevance score between zero and one. Higher similarity values indicate stronger alignment between candidate profiles and job requirements.

5. SYSTEM ARCHITECTURE

The proposed system architecture is designed as a modular and scalable pipeline that enables automated resume analysis and job recommendation while maintaining computational efficiency and deployment flexibility. The architecture integrates multiple processing layers that operate sequentially, beginning with resume ingestion and concluding with ranked job recommendations.

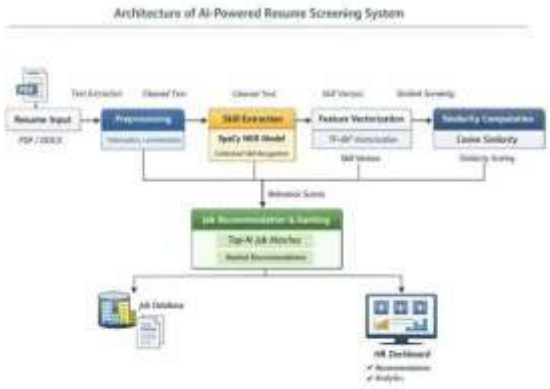
At the initial stage, the system accepts resumes in commonly used formats such as PDF and DOCX, ensuring compatibility with real-world recruitment workflows. These documents are processed through a document parsing mechanism that extracts raw textual content and converts it into a machine-readable format. The extracted text is then passed to a preprocessing layer where normalization procedures such as tokenization, stopword removal, lemmatization, punctuation filtering, and case standardization are performed to eliminate noise and ensure linguistic consistency. This preprocessing stage plays a critical role in improving the

reliability of downstream entity recognition and feature extraction processes.

Following preprocessing, the core analytical component of the architecture — the custom-trained SpaCy Named Entity Recognition model — is applied to identify skill entities within the resume text. The NER model operates by analyzing contextual relationships between words rather than relying solely on dictionary matching, thereby improving extraction robustness across varied resume structures and terminologies.

Once skill entities are extracted, they are structured and forwarded to the feature representation module, where the textual information is transformed into numerical vectors using the Term Frequency–Inverse Document Frequency weighting scheme. This transformation allows the system to quantify the relative importance of each skill within a candidate profile while minimizing the influence of overly common terms.

The similarity computation layer then compares the resume vector with predefined job description vectors using cosine similarity, producing a normalized relevance score that represents alignment between candidate competencies and job requirements. The ranking mechanism sorts these similarity scores in descending order and selects the top matching job roles as recommendations.



A. Mathematical Model

Let:

$R = \{r1, r2, r3, ..., rn\}$ be the set of candidate resumes

$J = \{j1, j2, j3, ..., jm\}$ be the predefined set of job descriptions

Each resume ri is parsed and structured into relevant fields as:

$ri = \{name, contact, skills, education, experience\}$ ...(1)

B. Resume Parsing

Each resume ri is uploaded in PDF or DOCX format through the user interface. The document parsing module extracts raw text from the file and performs preprocessing operations such as tokenization, stopword removal, lemmatization, and normalization.

The extracted information includes:

- Name: Candidate's full name

- Contact: Email and phone number

- Skills: Technical and domain skills identified

- Education: Academic background

- Experience: Professional history

After extraction, the resume is structured and stored in JSON format for further processing. This structured representation ensures compatibility with downstream NLP modules.

Skill Extraction using SpaCy NER

A custom-trained SpaCy Named Entity Recognition (NER) model is employed to identify contextual skill entities from the pre-processed resume text.

$Si = NER(ri)$

where Si represents the extracted set of skills from resume ri.

Unlike rule-based keyword matching, the NER model captures contextual multi-word skills and domain-specific terminology.

TF-IDF Vector Representation

To numerically represent candidate profiles and job descriptions, TF-IDF vectorization is applied.

For each resume ri, a vector representation is generated:

$Vri = TFIDF(Si)$

Similarly, each job description jk is transformed into:

$Vjk = TFIDF(jk)$

The TF-IDF weighting scheme is defined as:

TFIDF(t, d) = TF(t, d) x log(N / DF(t)) ...(2)

where:

- TF(t, d) is the term frequency of term t in document d

- N is the total number of documents

- DF(t) is the number of documents containing term t

Cosine Similarity

To measure alignment between a resume and a job description, cosine similarity is computed. The similarity score Sik between resume ri and job jk is calculated as:

Sik = (Vri . Vjk) / (||Vri|| x ||Vjk||) ...(3)

where:

- Vri . Vjk is the dot product of the two vectors

- ||Vri|| and ||Vjk|| are the Euclidean norms of the respective vectors

The similarity score lies in the range:

0 <= Sik <= 1

Higher values indicate stronger alignment between candidate skills and job requirements.

Job Recommendation and Ranking

For each resume, similarity scores are computed across all job descriptions. The system ranks jobs in descending order of similarity:

Rank(ri) = sort(Si1, Si2, ..., Sim)

The top-N jobs with the highest similarity scores are recommended.

Threshold-Based Filtering

A predefined threshold τ is used to filter relevant matches:

If Sik >= τ => Candidate is recommended ...(4)

If Sik < τ => Candidate is not recommended

This automated filtering reduces manual screening effort.

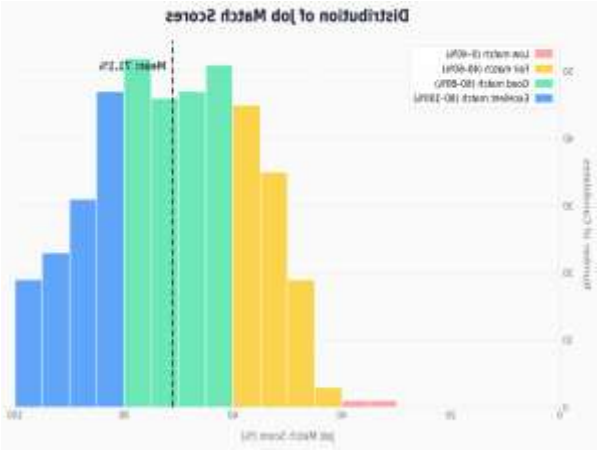
B. Data Flow

The data flow of the proposed AI-powered Resume Skill Extraction and Job Recommendation System is organized into three major phases: input, processing, and output.

In the input phase, resumes are uploaded through the user interface in PDF or DOCX format. The document parsing module extracts raw textual data and converts it into a structured format.

In the processing phase, the extracted text undergoes preprocessing followed by contextual skill extraction using the SpaCy NER model. The identified skills are transformed into TF-IDF vectors, and cosine similarity is computed between resume vectors and job description vectors. The similarity scores are then sorted to rank job roles according to relevance.

In the output phase, the system generates ranked job recommendations for the candidate. The results are displayed through the interface and can also be exported in structured JSON format for integration with HR systems. This structured pipeline ensures scalability, efficiency, and interpretability while minimizing manual intervention in the recruitment process.



6. IMPLEMENTATION

The proposed AI-Powered Resume Skill Extraction and Job Recommendation System is implemented using a modular and scalable architecture with a clear separation between backend processing, API management, and frontend visualization. The system leverages Python as

the primary development language due to its rich ecosystem of Natural Language Processing (NLP), machine learning, and web development libraries. The architecture ensures efficient resume parsing, contextual skill extraction,

similarity computation, and job recommendation generation.

Algorithm: Resume Screening and Job Recommendation Process

Input: Resume ri, Job Description jk

Output: Ranked job recommendations

1: Parse ri using document parser to extract text

2: Preprocess text (tokenization, stopword removal, lemmatization)

3: Extract skills using SpaCy NER: Si = NER(ri)

4: Generate TF-IDF vectors: Vri, Vjk

5: Compute cosine similarity: Sik = (Vri . Vjk) / (||Vri|| x ||Vjk||)

6: Rank jobs based on similarity scores

7: if Sik >= τ then

8: Recommend job role

9: else

10: Do not recommend

11: end if

12: Display ranked results on dashboard

A. Backend Processing

The backend is developed entirely in Python and handles resume parsing, preprocessing, skill extraction, vectorization, similarity computation, and recommendation ranking. The system integrates multiple libraries to ensure modularity and efficiency.

The resume parsing module extracts textual content from PDF and DOCX resumes and converts it into a structured format. The preprocessing pipeline includes tokenization, normalization, stopword removal, punctuation filtering, and lemmatization to ensure consistent linguistic representation.

The core skill extraction mechanism utilizes SpaCy's Named Entity Recognition (NER) framework. A custom-trained NER model identifies domain-specific skill entities from resumes. Unlike traditional keyword-based approaches, the contextual NER model captures multi-word technical skills and improves extraction robustness.

For numerical representation, the system employs the TF-IDF vectorization technique to transform textual data into weighted feature vectors. This method ensures that frequently occurring but less informative terms are assigned lower weights, while important domain-specific skills receive higher importance.

Cosine similarity from scikit-learn is used to measure alignment between resume vectors and job description vectors. The similarity score serves as the ranking metric for job recommendations.

All processed data is stored in structured JSON format to ensure lightweight storage and easy integration with other systems.

B. API Development

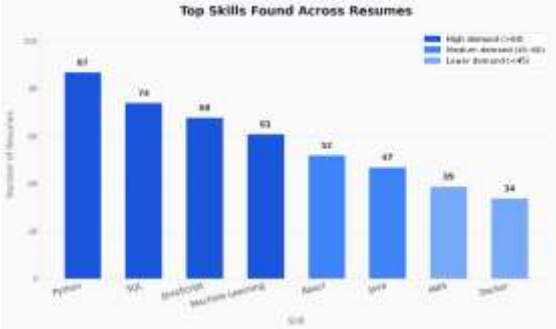
The system backend functionalities are exposed through RESTful APIs built using FastAPI. FastAPI is chosen due to its high performance, asynchronous support, and automatic API documentation generation.

The API layer handles resume upload requests, initiates parsing and processing tasks, computes similarity scores, and returns ranked job recommendations. The framework ensures secure communication between frontend and backend modules while maintaining high-speed request handling and scalability.

C. Frontend Interface

The frontend consists of two major components: the Candidate Interface and the HR Dashboard.

The Candidate Interface allows users to upload resumes and view recommended job roles based on similarity scores. The interface provides real-time validation and ensures smooth interaction.The HR Dashboard displays extracted skills, similarity scores, and ranked job recommendations in an interactive format. Visualization components present similarity distributions and allow HR personnel to filter candidates based on threshold values.



REFERENCES

[1] N. Ali, Z. H. Khand, J. Ahmed, and G. Mujtaba, "Resume classification system using natural language processing and machine learning techniques," Mehran University Research Journal of Engineering and Technology, 2021.
[2] M. Honnibal and I. Montani, "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing," To appear, 2017. (Foundational citation for the custom SpaCy NER model used in the project architecture).
[3] B. Kinge, S. Mandhare, P. Chavan, and S. M. Chaware, "Resume screening using machine learning and NLP: A proposed system," International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 2021.
[4] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011. (Foundational citation for the TF-IDF vectorization and Cosine Similarity modules used in the project).
[5] P. K. Roy, S. S. Chowdhary, and R. Bhatia, "A Machine Learning approach for automation of Resume Recommendation system," Procedia Computer Science, vol. 167, pp. 2318-2327, 2020.
[6] S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. O'Reilly Media, Inc., 2009. (Foundational citation for the NLTK preprocessing pipeline, including tokenization and stopword removal).
[7] S. Lokesh, S. Mano Balaje, E. Prathish, and B. Bharathi, "Resume screening and recommendation system using machine learning approaches," Computer Science & Engineering: An International Journal (CSEIJ), 2021.
[8] L. Cabrera-Diego, B. Durette, M. Lafon, J.-M. Torres-Moreno, and M. El-Bèze, "Ranking resumes automatically using only resumes: A method free of job offers," Expert Systems with Applications, vol. 123, pp. 91-107, 2019.
[9] F. A. Jafari and R. Simon, "Automated Resume Screening System Using NLP and Machine Learning," International Journal of Scientific Research in Engineering and Management (IJSREM), vol. 09, no. 05, 2025.
[10] M. Saatçı, R. Kaya, and R. Ünlü, "Resume screening with natural language processing (NLP)," Alphanumeric Journal, 2021
[11] Roy, P. K., Chowdhary, S. S., & Bhatia, R. (2020). A machine learning approach for automation of resume recommendation system. Procedia Computer Science, 167, 2318–2326.
[12] Colucci, S., Noia, T. D., Di Sciascio, E., Donini, F. M., Mongiello, M., & Mottola, M. (2003). A formal approach to ontology-based semantic match of skills descriptions. Journal of Universal Computer Science, 9(12), 1437–1454.
[13] Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. O'Reilly Media.
[14] Honnibal, M., Montani, I., Van Landeghem, S., & Boyd, A. (2020). spaCy: Industrial-strength Natural Language Processing in Python. Zenodo. https://doi.org/10.5281/zenodo.1212303
[15] Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. Information Processing & Management, 24(5), 513–523.
[16] Streamlit Inc. (2023). Streamlit: A faster way to build and share data apps. https://streamlit.io