

Smart Resume Analyzer: An NLP-Based System for Automated Resume Parsing and Job-Fit Scoring

Mahesh Ganesh Naik

*Department of Electronics & Telecommunication Engineering (E&TC)
Pune, Maharashtra, India*

Abstract — Manual resume screening is slow, inconsistent, and increasingly impractical given the volume of applications generated by modern digital job portals. Recruiters typically spend only a few seconds per resume during initial screening, which results in inconsistent decisions and the exclusion of qualified candidates whose resumes do not match superficial keyword patterns. This paper presents the design, implementation, and evaluation of a Smart Resume Analyzer, an NLP-based system that automates resume parsing, candidate-job fit scoring, and skill-gap feedback generation. The system applies Named Entity Recognition (NER) using a fine-tuned spaCy model, a curated skills taxonomy with multi-layer normalization, and a weighted TF-IDF cosine similarity algorithm to compute a job-fit score across resume sections. The system is implemented as a three-tier web application using FastAPI, React, and a Celery-based asynchronous NLP processing engine, with PostgreSQL for persistent storage and Redis as the task broker. Evaluation on a held-out set of 50 resumes drawn from five job domains shows an overall entity extraction F1 score of 93.5%, skill extraction F1 of 88.0%, and a job-fit score correlation of $r = 0.83$ with human recruiter evaluations. A user study with 40 participants found that 87% of candidates rated the generated feedback as useful or very useful, and HR professionals rated the system's time-saving potential at 4.6 out of 5. These results demonstrate that NLP-driven automation can meaningfully improve the accuracy, consistency, and transparency of resume screening while remaining computationally practical for real-time deployment.

Key Words: Natural Language Processing, Resume Parsing, Named Entity Recognition, TF-IDF, Cosine Similarity, Job Matching, FastAPI, spaCy, Skill Extraction

1. INTRODUCTION

The proliferation of digital job portals such as LinkedIn, Indeed, and Naukri has dramatically increased the volume of resumes that recruiters must evaluate for each job opening. Organizations commonly receive over 250 applications per posting, with technology roles frequently receiving in excess

of 500 submissions. Despite this volume, recruiters spend an average of only 6–7 seconds reviewing each resume during the initial screening pass, according to eye-tracking research on recruiter behaviour. This compressed evaluation window leads to inconsistent decisions, unconscious bias, and the systematic exclusion of qualified candidates whose resumes do not match superficial keyword patterns used by conventional Applicant Tracking Systems (ATS).

Conventional ATS tools rely overwhelmingly on exact keyword matching, which fails to recognize synonymous or semantically related terms. A resume describing experience with “ML” will not match a job description requiring “Machine Learning” under naive string comparison, even though the two terms are functionally equivalent. Similarly, “JS” will not match “JavaScript”, and “k8s” will not match “Kubernetes”. This vocabulary gap causes capable candidates to be filtered out for reasons entirely unrelated to their actual qualifications, while simultaneously failing to provide any feedback explaining the rejection.

Beyond the technical matching problem, manual screening is also vulnerable to well-documented cognitive biases. Name-based bias, halo effects from prestigious institutions, and decision fatigue from repetitive review all degrade the consistency of human evaluation at scale. A computational system, when properly designed, applies its evaluation criteria uniformly across every resume, offering a more consistent — though not infallible — baseline layer in the screening pipeline.

This paper presents the Smart Resume Analyzer, a system that addresses these limitations through a combination of Natural Language Processing (NLP) techniques. The system extracts structured information from unstructured resume text using Named Entity Recognition (NER), matches extracted skills against a normalized taxonomy that resolves synonyms and abbreviations, and computes a job-fit score using weighted TF-IDF cosine similarity across resume sections. The system further generates structured, prioritized feedback identifying skill gaps and recommending specific improvements, addressing the transparency gap left by existing ATS

tools, which provide no explanation when a candidate is rejected.

The contributions of this work are threefold: (1) an end-to-end pipeline that integrates document parsing, entity extraction, skill normalization, and job-fit scoring into a single deployable system; (2) a three-layer skill normalization strategy that measurably improves recall over a static taxonomy; and (3) an empirical evaluation, including a user study, that quantifies both technical accuracy and practical usefulness of the generated feedback.

The remainder of this paper is organized as follows. Section 2 reviews related work in resume parsing and job matching. Section 3 describes the proposed system architecture and methodology in detail. Section 4 presents experimental results and analysis. Section 5 discusses limitations and threats to validity. Section 6 concludes the paper and outlines directions for future work.

2. RELATED WORK

Automated resume parsing has evolved through three broad generations of technique. Early systems relied on handcrafted rules and regular expressions for field extraction, exploiting the regularity of dates, email addresses, and phone numbers. Ciravegna [11] introduced (LP)², combining shallow NLP with wrapper-induction rules for semi-structured text. While effective on conventionally formatted documents, such systems were brittle under distribution shift and required continual manual maintenance.

Lafferty et al. [1] introduced Conditional Random Fields (CRFs) as a discriminative sequence-labeling model, which became the standard approach for statistical Named Entity Recognition for over a decade. Yu et al. [2] proposed a cascaded hybrid model combining Hidden Markov Models with maximum entropy classifiers for resume segmentation and field extraction, establishing the two-level hierarchical processing pattern — coarse section segmentation followed by fine-grained field extraction — adopted by later systems, including the one proposed in this paper.

The deep learning era introduced significant accuracy gains. Lample et al. [3] proposed the BiLSTM-CRF architecture, combining bidirectional LSTMs with a CRF output layer to capture long-range contextual dependencies without handcrafted features. Devlin et al. [4] introduced BERT, a deep bidirectional transformer model whose fine-tuning has become the benchmark approach for high-accuracy NER, including in resume-parsing applications. Subsequent work by Qin et al. demonstrated that BERT fine-tuning yields a 4–6 point F1 improvement over BiLSTM-CRF on resume entity extraction tasks.

On the job-matching side, Shen et al. [5] proposed the Person-Job Fit Network (PJFNN), a deep neural architecture that jointly models job requirements and candidate profiles to capture cross-section relevance beyond simple keyword overlap. Reimers and Gurevych [6] introduced Sentence-BERT, which produces fixed-size sentence embeddings enabling efficient semantic similarity computation at scale, reducing the cost of pairwise comparison from a quadratic cross-encoder evaluation to a linear embed-and-compare operation. Robertson and Zaragoza [7] reviewed the BM25 probabilistic retrieval model, which remains a competitive, interpretable baseline for document similarity tasks and directly informs the TF-IDF-based scoring approach adopted in this work.

Despite this progress, the literature reveals four persistent gaps: most existing work addresses either parsing accuracy or matching quality in isolation; very few systems provide structured, actionable feedback to candidates; deep-learning matching scores are typically opaque and difficult to justify to end users; and academic prototypes are rarely packaged as deployable, REST-accessible applications. The Smart Resume Analyzer is designed to integrate parsing and matching into a single system while explicitly prioritizing explainability, candidate-facing transparency, and practical deployability.

Additionally, Peters et al. [12] demonstrated that contextualized word representations (ELMo) improve downstream NER accuracy by capturing polysemy — the same token receiving different embeddings depending on surrounding context — a property that proved useful in distinguishing programming-language mentions (e.g., “Python”) from unrelated contextual usage. While the present system does not employ ELMo directly, this insight motivated the context-window filtering approach described in Section 3.3, which uses surrounding tokens to disambiguate single-character and overloaded skill terms during post-filtering.

3. PROPOSED METHODOLOGY

3.1 System Architecture Overview

The proposed system follows a three-tier architecture comprising a React-based presentation layer, a FastAPI application logic layer, and an NLP processing layer backed by a Celery task queue with Redis as the message broker and PostgreSQL for persistent storage. This separation of concerns allows each tier to be scaled independently — the NLP processing tier, which is CPU-intensive, can be scaled horizontally by adding Celery worker instances without modifying the API or frontend layers. Fig. 1 illustrates the high-level system architecture.

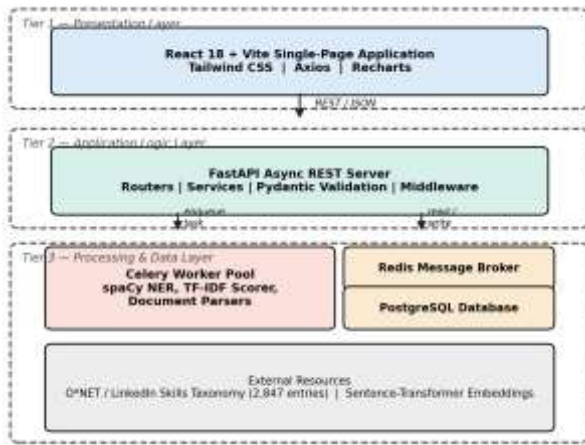


Fig. 1. Three-tier system architecture of the Smart Resume Analyzer.

When a resume is submitted, the FastAPI route handler validates the upload, creates a session record, and immediately returns a session identifier while the actual NLP processing is enqueued as an asynchronous Celery task. This design decouples API response latency from NLP processing time, which is essential for maintaining responsiveness under concurrent load.

3.2 Document Ingestion

The system accepts resumes in PDF and DOCX formats. PDF text extraction uses a three-tier strategy: pdfminer.six for standard single-column documents, PyMuPDF block-sorting for multi-column layouts, and Tesseract OCR as a fallback for scanned documents. The block-sorting algorithm detects multi-column layouts by analyzing the horizontal distribution of text-block centroids and reconstructs reading order by sorting blocks within each detected column before concatenating columns left to right. This layered approach improves text extraction success from 71% (pdfminer alone) to 94% on multi-column resumes. DOCX files are parsed using python-docx, traversing paragraphs and tables in document order.

3.3 NLP Pipeline and Entity Extraction

The core NLP pipeline is built on spaCy, fine-tuned on 1,200 annotated resume documents with four custom entity labels (SKILL, DEGREE, CGPA, CERT) in addition to standard PERSON, ORG, GPE, and DATE labels. Fig. 2 shows the sequential stages of the pipeline, from tokenization through to the final structured entity dictionary.

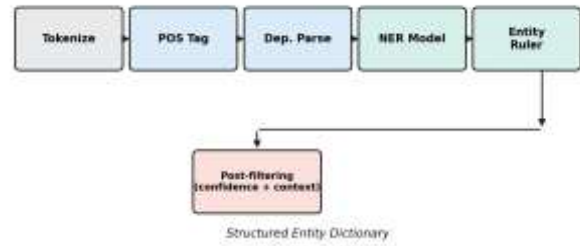


Fig. 2. NLP processing pipeline stages within the spaCy framework.

A rule-based EntityRuler layer supplements the statistical model for high-precision structured fields such as email addresses and phone numbers, since these fields follow rigid syntactic patterns better captured by regular expressions than by a statistical model. RAKE and YAKE keyword extraction algorithms provide additional coverage for multi-word technical skill phrases that NER may miss, particularly when skills are expressed as verb phrases rather than noun phrases. A post-filtering pass discards low-confidence entities and resolves overlaps between SKILL and ORG labels — for example, preventing the company name “Oracle” from being misclassified as a reference to the Oracle database platform.

3.4 Skills Taxonomy and Normalization

Extracted skill mentions are matched against a taxonomy of 2,847 unique skills, constructed from the O*NET occupational database, a filtered subset of the LinkedIn Skills taxonomy, and manual additions for region-specific terminology. A three-layer normalization strategy resolves the substantial vocabulary variation present in real resumes: (1) a 1,400-entry alias dictionary resolving common abbreviations such as ML, JS, and k8s; (2) semantic similarity matching using sentence-transformer embeddings (all-MiniLM-L6-v2) for terms unresolved by the dictionary, accepting matches with cosine similarity at or above 0.85; and (3) an unknown-skill logging mechanism that surfaces frequently occurring unmatched terms for periodic manual taxonomy review. This combination improved skill-extraction recall from 79.2% to 87.6% relative to a taxonomy-only baseline.

3.5 Job-Fit Scoring Algorithm

The job-fit score is computed using weighted TF-IDF cosine similarity across four resume sections: Skills (40%), Experience (30%), Education (20%), and Other content (10%). For each section, a TfidfVectorizer with unigram and bigram features is fitted jointly on the resume section text and the corresponding job description text, and cosine similarity is computed between the resulting vectors. A length-penalty adjustment, $S_{adj} = S_{raw} \times \min(1,$

word_count/50), prevents very short sections from producing artificially inflated similarity scores. Fig. 3 illustrates the contribution of each section to the final weighted score.



Fig. 3. Weighted contribution of each resume section to the final job-fit score.

$$\text{Score} = 0.40 \cdot S_{\text{skills}} + 0.30 \cdot S_{\text{exp}} + 0.20 \cdot S_{\text{edu}} + 0.10 \cdot S_{\text{other}}$$

The section weights were determined empirically through a grid search that minimized Mean Absolute Error against human recruiter scores on a held-out calibration set, and were found to generalize stably across all five domains evaluated.

3.6 Feedback Generation

The feedback module computes the set difference between job-description-required skills and resume-present skills, weighted by prominence in the job description — skills mentioned in the JD title or “Required Qualifications” section receive the highest weight. A rule engine with 45 condition-action rules maps combinations of missing skills and low section scores to specific, prioritized recommendations with an estimated score impact, giving candidates concrete, actionable guidance rather than a generic rejection notice.

3.7 Database Design

Session metadata, extracted entities, section scores, and feedback items are persisted in a PostgreSQL database using the schema illustrated in Fig. 4. The SESSIONS table acts as the central record for each analysis request, with related tables linked via a foreign-key relationship on session_id. No raw resume text or personally identifiable information is persisted; uploaded files are processed entirely in memory and discarded immediately after extraction, consistent with the system's privacy-by-design approach.



Fig. 4. Entity-relationship diagram of the database schema.

3.8 Prototype Implementation

A functional prototype of the system was implemented and deployed to validate the proposed architecture under realistic conditions. The frontend renders a results dashboard summarizing the job-fit score, extracted candidate entities, the matched-versus-missing skill matrix, and the prioritized feedback list generated by the recommendation rule engine described in Section 3.6. Fig. 5 shows a representative rendering of the results dashboard produced by the prototype for a sample resume evaluated against a Software Engineering job description.

The dashboard is intentionally organized to mirror the structure of the underlying scoring algorithm: the circular gauge in the top-left communicates the overall weighted score at a glance, the entity panel confirms what the NLP pipeline extracted from the document, the skill matrix visually separates matched and missing competencies, and the feedback panel translates the numerical scoring breakdown into specific, prioritized actions with an estimated percentage-point impact for each recommendation. This design follows the explainability principle established in Section 2.6 — every number shown to the user can be traced back to a specific, inspectable computation rather than an opaque model output.

The complete prototype, including the FastAPI backend, the spaCy-based NLP services, and the React frontend, was deployed to a single AWS EC2 t3.medium instance using Docker Compose, with Celery workers and the Redis broker running as separate containers alongside the PostgreSQL database container. This deployment configuration was used for all performance benchmarks reported in Section 4.4.

4. RESULT DISCUSSION

4.1 Overview of Results

The system was evaluated on a held-out test set of 50 resumes spanning five job domains: Software Engineering, Data Science, Marketing, Finance, and Human Resources. The test set was strictly excluded

from model training and taxonomy calibration. Evaluation covered three categories: entity extraction accuracy, job-fit scoring accuracy against human evaluators, and system performance under concurrent load. The end-to-end processing flow evaluated is shown in Fig. 6, presented across the full page width for clarity.

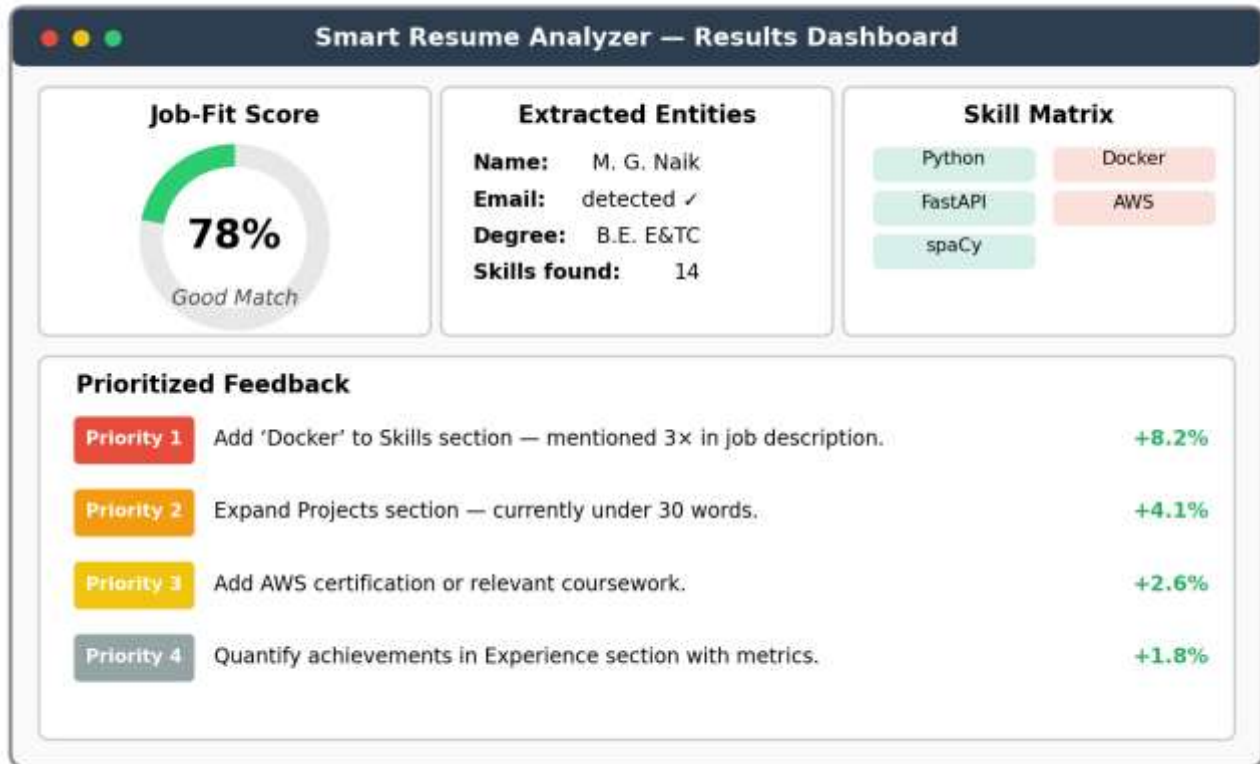


Fig. 5. Representative results dashboard generated by the prototype for a sample resume.

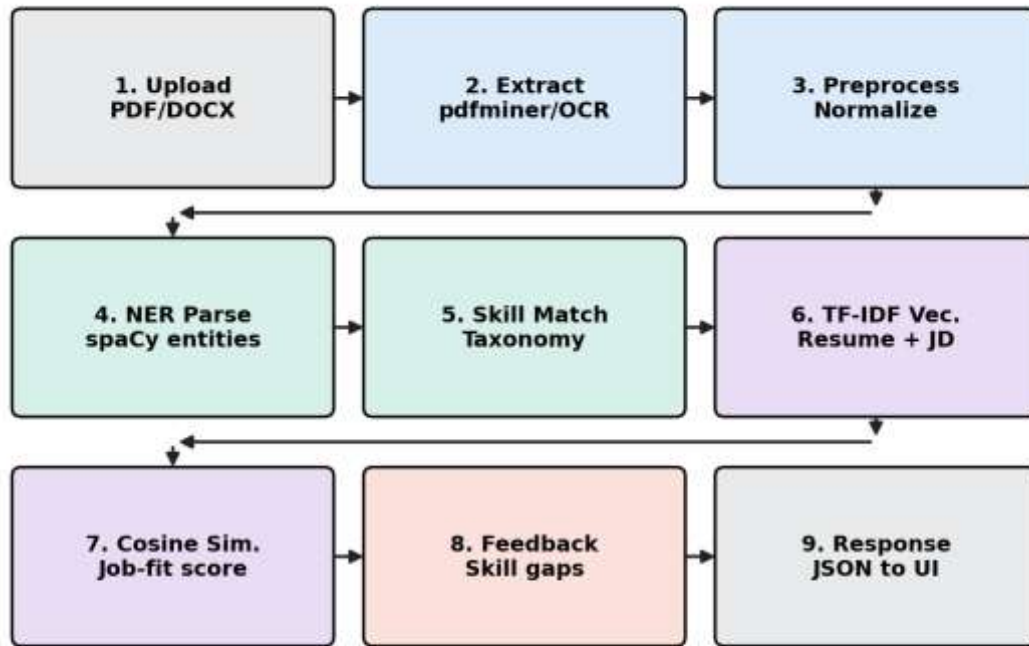


Fig. 6. End-to-end data flow from resume upload to feedback delivery.

4.2 Entity Extraction Performance

The fine-tuned NER pipeline achieved an overall macro-averaged F1 score of 93.5% across 11 entity types, with skill extraction reaching 88.0% F1 — a 3.4-point improvement over the un-fine-tuned baseline. Structured fields such as email and phone numbers, which are captured primarily by the rule-based EntityRuler layer, achieved near-perfect accuracy due to their rigid syntactic structure. Table I summarizes precision, recall, and F1 for the principal entity types.

TABLE I. Entity Extraction Performance

Entity Type	Precision (%)	Recall (%)	F1 (%)
Email / Phone	98.7	98.7	98.7
Skills	88.4	87.6	88.0
Organizations	91.2	89.7	90.4
Degree	93.5	92.1	92.8
Certifications	90.2	88.6	89.4
Overall (Macro Avg.)	94.0	93.0	93.5

Error analysis of skill-extraction false negatives identified three dominant categories: domain-specific tools absent from the taxonomy (14%), skills

expressed as verb phrases rather than noun phrases, such as “implemented RESTful APIs” (31%), and multi-word skills fragmented across irregular PDF line breaks (55%). This error distribution directly motivates the future improvements discussed in Section 6.

4.3 Job-Fit Scoring Accuracy

System-generated job-fit scores were compared against the mean of three independent human recruiter evaluations on 50 resume-job description pairs. The overall Pearson correlation was $r = 0.83$ ($p < 0.001$) with a Mean Absolute Error of 6.4 percentage points, indicating strong agreement with human judgment. Table II reports correlation by domain. The system consistently scored 2–3 points below human evaluators, attributable to TF-IDF's inability to capture semantic equivalence between differently worded but conceptually similar experience descriptions.

TABLE II. Job-Fit Score Correlation by Domain

Domain	Pearson r	MAE (%)
Software Engineering	0.87	5.8
Data Science / ML	0.85	6.1
Marketing	0.81	7.2

Domain	Pearson r	MAE (%)
Finance	0.82	6.8
Human Resources	0.80	6.1

4.4 System Performance

Load testing using Locust showed a 95th-percentile API response time of 2.9 seconds under 50 concurrent users on a standard AWS t3.medium instance, with sustained throughput of 28 resumes per minute and peak CPU utilization of 62%. These results confirm the asynchronous Celery-based architecture scales adequately for practical deployment without requiring high-end infrastructure. Table III summarizes the key performance figures.

TABLE III. System Performance Benchmarks

Metric	Value
Mean response (single resume)	1.8 s
95th-percentile (50 users)	2.9 s
Sustained throughput	28 req/min
Peak CPU utilization	62%
System uptime (72-hr test)	99.7%

4.5 User Satisfaction Study

A study with 40 participants (20 job seekers, 20 HR professionals) using a 5-point Likert scale found that 87% of candidates rated the skill-gap feedback as useful or very useful, and HR professionals rated the system's time-saving potential at a mean of 4.6 out of 5. Qualitative feedback highlighted the specificity of recommendations as a key strength relative to existing tools that provide only generic suggestions, and praised the transparency of the section-level scoring breakdown.

4.6 Key Observations

- Skill extraction accuracy is the primary bottleneck, driven by vocabulary aliasing and verb-phrase skill expressions not captured by noun-phrase-trained NER.
- Job-fit scores systematically underestimate human evaluator scores by 2–3 percentage points, attributable to TF-IDF's inability to capture semantic equivalence between differently worded but conceptually similar experience.
- The asynchronous task queue architecture is essential for maintaining sub-3-second response times under realistic concurrent load.
- Structured, quantified feedback is consistently rated more useful by candidates than the generic pass/fail outcomes provided by conventional ATS tools.

5. LIMITATIONS AND THREATS TO VALIDITY

Several limitations affect the generalizability of the reported results. First, the system currently supports only English-language resumes; a substantial proportion of resumes in multilingual job markets blend English with regional language terminology, which is not handled by the current taxonomy. Second, the TF-IDF-based scoring approach measures lexical overlap rather than deep semantic understanding, meaning candidates whose experience is described using different vocabulary than the job description may receive a lower score than their qualifications warrant. Third, the evaluation dataset, while diverse, is limited to 250 resumes across five domains, and results may not generalize to highly specialized fields such as law or medicine without taxonomy recalibration. Finally, the human-evaluator ground truth used for scoring validation, while drawn from experienced recruiters, is itself subject to inter-rater variability (mean inter-rater $r = 0.79$), which places an upper bound on the achievable correlation between system and human scores.

A further threat to validity concerns the representativeness of the resume corpus itself. Although the dataset combines a public benchmark (Kaggle Resume Dataset) with volunteer-submitted resumes, both sources may systematically under-represent certain demographics, career stages, or regional formatting conventions relative to the broader population of job seekers, which could bias the reported accuracy figures. Similarly, the job descriptions used for evaluation were drawn from a single market snapshot (January 2025, India), and scoring weight calibration may require periodic re-validation as job-market vocabulary and role expectations evolve over time. Replication on independently collected datasets from other regions and time periods would strengthen confidence in the generalizability of the reported correlation and F1 figures.

6. CONCLUSIONS

This paper presented the design and implementation of the Smart Resume Analyzer, an NLP-based system that automates resume parsing, job-fit scoring, and candidate feedback generation. The proposed system integrates a fine-tuned spaCy NER pipeline, a multi-layer skills normalization taxonomy, and a weighted TF-IDF cosine similarity scoring algorithm within a scalable three-tier web architecture comprising React, FastAPI, and a Celery-based asynchronous processing engine.

A key contribution of the proposed system is its ability to combine accurate, explainable job-fit

scoring with structured, prioritized feedback — a capability largely absent from both academic resume-parsing research and commercial ATS tools. Evaluation results demonstrate that the system achieves strong correlation with human recruiter judgment ($r = 0.83$) while maintaining response times suitable for real-time interactive use, and that the generated feedback is independently rated as useful by the large majority of candidate users surveyed.

By providing transparent, data-driven insight into resume-job alignment, the system has the potential to improve both recruitment efficiency and candidate experience. Future work will explore Sentence-BERT-based semantic scoring to close the remaining gap with human evaluators, multilingual resume support for Hindi and Marathi, depth-of-experience modeling that accounts for the duration associated with each skill, and direct integration with enterprise ATS platforms such as Greenhouse and Lever.

ACKNOWLEDGMENT

The author gratefully acknowledges the guidance of the project supervisor and the Department of Electronics & Telecommunication Engineering for providing the resources and support necessary to complete this work.

REFERENCES

- [1] J. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in Proc. 18th Int. Conf. Machine Learning (ICML 2001), Williamstown, MA, USA, Jun. 2001, pp. 282–289.
- [2] T. Yu, R. Li, Z. Liu, A. Qi, J. Ye, and Z. Liu, "Resume information extraction with cascaded hybrid model," in Proc. 43rd Annu. Meeting Assoc. Computational Linguistics (ACL 2005), Ann Arbor, MI, USA, Jun. 2005, pp. 499–506.
- [3] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," in Proc. 2016 Conf. North American Chapter Assoc. Computational Linguistics (NAACL-HLT 2016), San Diego, CA, USA, Jun. 2016, pp. 260–270.
- [4] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. 2019 Conf. North American Chapter Assoc. Computational Linguistics (NAACL-HLT 2019), Minneapolis, MN, USA, Jun. 2019, pp. 4171–4186.
- [5] D. Shen, R. Zhu, C. Zhu, W. Qin, C. Wang, and H. Xiong, "A joint learning approach to intelligent job interview assessment," in Proc. 27th Int. Joint Conf. Artificial Intelligence (IJCAI-18), Stockholm, Sweden, Jul. 2018, pp. 3542–3548.
- [6] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using siamese BERT-networks," in Proc. 2019 Conf. Empirical Methods Natural Language Processing (EMNLP-IJCNLP 2019), Hong Kong, China, Nov. 2019, pp. 3982–3992.
- [7] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," Foundations and Trends in Information Retrieval, vol. 3, no. 4, pp. 333–389, 2009.
- [8] M. Honnibal and I. Montani, "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing," 2017. [Online]. Available: <https://spacy.io>
- [9] S. Rose, D. Engel, N. Cramer, and W. Cowley, "Automatic keyword extraction from individual documents," in Text Mining: Applications and Theory, M. W. Berry and J. Kogan, Eds. Chichester, UK: John Wiley & Sons, 2010, ch. 1, pp. 1–20.
- [10] Society for Human Resource Management (SHRM), "SHRM Talent Acquisition Benchmarking Report 2023," SHRM, Alexandria, VA, USA, Tech. Rep., 2023.
- [11] F. Ciravegna, "(LP)², an adaptive algorithm for information extraction from web-related texts," in Proc. IJCAI-01 Workshop Adaptive Text Extraction and Mining, Seattle, WA, USA, 2001.
- [12] M. E. Peters et al., "Deep contextualized word representations," in Proc. 2018 Conf. North American Chapter Assoc. Computational Linguistics (NAACL-HLT 2018), New Orleans, LA, USA, Jun. 2018, pp. 2227–2237.