

SmartResume AI: An Intelligent Resume Analysis System Using Machine Learning and Natural Language Processing

Mr. Rajashekhar G C¹Director, Bhavana K S²Student 4th,

Rakshitha N H³Student 4th, Dhanusha R⁴Student 4th

Department of Computer Applications (MCA)

GM University, Davanagere

Abstract

Manual resume screening is slow and not consistent. Most systems used by recruiters rely on matching keywords that do not understand the meaning behind a candidate's experience. Existing tools give a score without explaining how it was calculated and do not provide a way to improve. These tools also treat resume analysis as a task, not connected to the rest of a candidate's job search. The SmartResume AI platform combines Natural Language Processing and machine learning to analyze resumes. It gives a score that shows how compatible a resume is with a job, a score, for the quality of the resume and identifies skills that are missing. It also compares resumes generates interview questions recommends jobs and has a career assistant that can have conversations.

The system uses Named Entity Recognition to parse resumes Sentence-BERT embeddings to match skills and a regression model to give scores. A career assistant is powered by a generation component. A working prototype has been. This paper explains its design and algorithms. The next step is to evaluate the system against tools using a large dataset. The main contribution of this work is to bring scoring, matching and career services that are currently separate. This work uses Resume Analysis, Applicant Tracking System, Natural Language Processing, Machine Learning and Explainable AI to make a resume analysis tool.

SmartResume AI uses Semantic Matching, Named Entity Recognition, Sentence-BERT, Job Recommendation and Career Assistant to help candidates. This system will help recruiters and candidates by making the resume screening process faster and more consistent.

Keywords — Resume Analysis, Applicant Tracking System, Natural Language Processing, Machine Learning, Explainable AI, Semantic Matching, Named Entity

Recognition, Sentence-BERT, Job Recommendation, Career Assistant

I. INTRODUCTION

The way companies hire people has changed a lot over the ten years. Many big companies use a system to look at resumes before a real person sees them. This system is called an Applicant Tracking System. It helps the people who hire by reducing the amount of work they have to do. It also makes it harder for people to get a job. Sometimes people do not get a job because their resume is not written in a way that the system can understand. It is not because they are not good enough for the job. This has made a lot of people want to find ways to help job seekers make their resumes work better with these systems. At the time computers have gotten better at understanding what people write. They can now tell what a resume means not just look for words. For example if someone says they led a team of engineers the computer should know that means they have leadership skills. Even if the resume does not say the word "leader" the computer should still understand.

Most of the tools that check resumes are not very good. They just look for words and give a score. They do not explain what the score means or how to make it better. We made a system called SmartResume AI. It looks at resumes in a way that makes sense and gives feedback that's helpful. It does not just give a score it tells you what you need to do to make your resume better.

The rest of this paper talks about what we did. We looked at what other people have written about this topic. We found out what was missing and what we wanted to do. We explain how we did it and what we used to make the system work. We also talk about what we found out when we tested it and how it is better than tools. We think this system is a way to help people get jobs and we want to keep making it better.

Section II looks at what other people have written, Section III talks about what's missing, Sections IV and V say what the problem is and what we want to do, Section VI explains how we did it, Section VII shows pictures of how the system works,

Section VIII talks about the computer programs we used. Section IX says what we found out when we tested it, Section X compares our system to tools, Sections XI-XIII talk about why our system is good what we think and what we want to do next.

II. LITERATURE SURVEY

A substantial body of recent work has examined different facets of automated resume analysis, ranging from parsing accuracy to semantic job matching. This section reviews ten representative studies published largely between 2023 and 2024 that inform the design of the proposed system.

Liang et al. [1] proposed a multi-task deep learning architecture that performs resume segmentation and named-entity recognition jointly within a single model, rather than as two separate pipeline stages. By sharing representations across both tasks, their approach reduced processing latency and improved entity-extraction accuracy for both English and Chinese resumes, demonstrating the benefit of unified parsing architectures over traditional two-stage rule-based pipelines.

Gopinath et al. [2] developed a convolutional neural network for resume parsing and job-role prediction, benchmarking it against Random Forest, Support Vector Machine, and BERT-based classifiers. Their system additionally used Word2Vec embeddings with cosine similarity to estimate the compatibility between a resume and a target job description, providing an early example of embedding-based rather than purely lexical matching — though the feedback generated remained largely score-only, without structured explanation.

Zhang et al. [3] introduced a decision-tree-based resume recommendation method for large employment platforms, applying cluster analysis during preprocessing to group candidates with similar profiles before recommendation. While computationally efficient and interpretable, decision-tree recommenders of this kind struggle to capture the fine-grained semantic nuance present in free-text resume descriptions, motivating the need for transformer-based alternatives for skill-level matching.

Warusawithana et al. [4] presented a layout-aware resume parser that combines NLP with rule-based heuristics to extract section-wise content while explicitly accounting for the visual structure of a resume document. Their system improved extraction completeness compared to entity-recognition-only approaches, highlighting that document-layout cues are an important but frequently overlooked signal in resume-parsing pipelines.

Ransing et al. [5] built a two-level stacked ensemble for resume screening, combining K-Nearest Neighbors, Linear Support Vector Classification, and XGBoost to rank candidates against job profiles. The stacked design improved ranking accuracy over any individual base learner, illustrating the value of ensemble methods for the ATS-scoring problem, although the study did not address the interpretability of the resulting rankings.

Spoorthi et al. [6] proposed an ensemble deep-learning model capable of classifying resumes supplied in multiple document formats, including Word and PDF, based on the skill sets referenced in the document. Their system emphasized robustness to formatting variation — a practical concern that is often underestimated in academic resume-parsing datasets that assume clean, uniformly structured input.

Bhatt et al. [7] designed an AI-driven resume analyzer that integrates a MapReduce-based preprocessing stage with a K-Nearest Neighbors classifier for job-role prediction, additionally generating personalized feedback intended to help candidates tailor their resumes to specific roles. This work is among the closer precursors to the present study in its emphasis on actionable feedback, though it does not incorporate transformer-based semantic matching or an explainability layer for the score.

Bharadwaj et al. [8] examined resume screening through a combination of NLP preprocessing and Long Short-Term Memory (LSTM) networks, showing that sequence models can capture contextual dependencies within resume text better than bag-of-words baselines. Their results reinforce the broader shift in the field toward deep contextual models — a trend that the present work extends further through transformer-based sentence embeddings.

Jaiswal et al. [9] investigated the use of BERT and XLNet transformer models to predict professional personality traits directly from resume text, demonstrating that pretrained language models can extract latent behavioral signals beyond

explicit skill keywords. This finding supports the feasibility of using transformer embeddings not only for skill matching, but also for richer candidate-profile characterization within a resume-analysis pipeline

Garcia and Fernandez [10] proposed an intelligent ATS scoring framework that combines NLP with deep learning for both scoring and skill-gap analysis, representing one of the few studies to explicitly target explainability alongside scoring accuracy.

Their reported results indicate that incorporating structured skill-gap output alongside a numeric score meaningfully improves candidate satisfaction with the tool — directly motivating the explainable, multi-module design adopted in SmartResume AI.

III. RESEARCH GAP

The literature reviewed in Section II points to four recurring gaps. First, systems such as those in [2] and [5] output a single numeric score without decomposing it into interpretable sub-factors, so a candidate cannot tell which specific change would most improve the result. Even the explainability-oriented framework of Garcia and Fernandez [10] does not report attributing individual score contributions to specific resume features. Second, matching in several of the systems reviewed — including the decision-tree recommender of Zhang et al. [3] and the earlier keyword- and rule-based parsers — relies on exact or near-exact term overlap rather than contextual similarity, so a candidate who describes a skill using different but equivalent wording is scored as if that skill were absent. The transformer-based work of Jaiswal et al. [9] and Zhao et al. [11] shows that contextual embeddings are available for this purpose, but they have not been applied specifically to skill-gap scoring in the systems surveyed here. Third, none of the reviewed systems connect resume scoring to downstream career activities. Bhatt et al. [7] come closest by generating personalized feedback, but their pipeline stops short of job recommendation, resume-version comparison, or interview preparation. Fourth, none of the surveyed systems provide an interactive, conversational channel through which a candidate can question or explore their results; feedback in every case reviewed is delivered as a static, one-time report. SmartResume AI is motivated directly by these four gaps.

IV. PROBLEM STATEMENT

Job seekers currently have no single tool that both predicts ATS compatibility in an explainable way and extends that analysis into the practical activities that follow it — identifying and closing skill gaps, comparing resume drafts, finding suitable openings, and preparing for interviews. Existing tools address at most one or two of these needs at a time, forcing candidates to move between disconnected services and to accept unexplained scores at face value. The problem addressed in this work is therefore the design of a single pipeline that produces an explainable, semantically grounded resume assessment and carries that assessment through to actionable, personalized career support.

V. OBJECTIVES

The specific objectives of this work are as follows. (1) To design an NLP-based resume parsing module that extracts structured entities — skills, education, and experience — from resumes supplied in common document formats. (2) To develop an ATS-score prediction model whose output is decomposed into human-readable, feature-level contributions rather than a single opaque number. (3) To construct a resume-quality score, computed independently of ATS compatibility, that reflects writing clarity and the use of quantifiable achievements. (4) To implement skill-gap analysis using contextual sentence embeddings so that semantically equivalent skills are recognized even when worded differently. (5) To support structured comparison between two or more resume versions. (6) To rank candidate-relevant job postings using the same semantic representation used for skill matching. (7) To generate interview questions grounded in the specific content of a candidate's resume, rather than generic role-based question banks. (8) To provide a conversational assistant that can answer follow-up questions about a candidate's own analysis results.

VI. PROPOSED METHODOLOGY

SmartResume AI is organized as a modular pipeline in which each stage consumes the structured output of the previous stage and produces additional derived features. The overall approach is deliberately explainable-by-design: every score produced by the system is backed by an intermediate, human-readable representation, rather than being emitted directly from an opaque model.

A. System Architecture

The system follows a layered architecture comprising a presentation layer, a processing layer, and a data layer, as illustrated in Fig. 1. The presentation layer exposes the web and mobile client interface through which resumes and job descriptions are submitted. The processing layer hosts the resume parser, the NLP and ML processing engine, the recommendation and scoring engine, the explainability module, the AI career assistant, and the interview-question generator. The data layer maintains the resume repository, job-description database, a curated skill ontology, and a vector store used for fast semantic similarity search.

Fig. 1. System Architecture of SmartResume AI

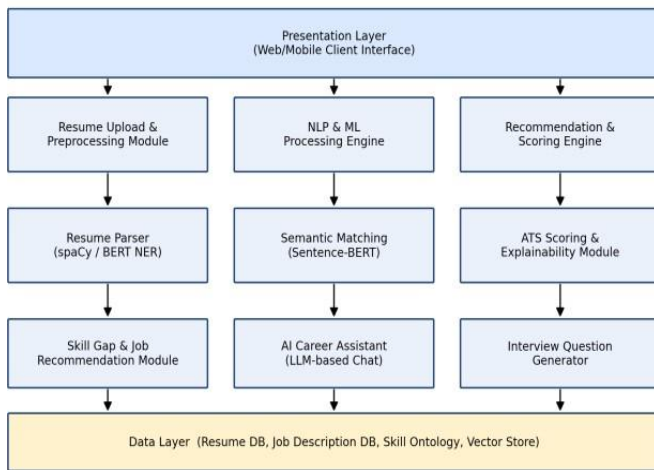


Fig. 1. Layered system architecture of SmartResume AI.

B. Workflow

The end-to-end workflow, shown in Fig. 9, begins with resume ingestion in PDF or DOCX format, followed by text cleaning and normalization, NLP-based feature extraction, ML-based scoring, semantic skill matching against the target job description, and finally recommendation generation, before results are delivered to the user through the dashboard.

C. Resume Parsing

Incoming resumes are first converted to plain text using format-specific extractors — PDFMiner for PDF and python-docx for DOCX. Layout cues such as section headers, bullet structure, and whitespace boundaries are then used to segment the document into candidate sections (e.g., Education, Experience, Skills, and Projects). A fine-tuned transformer-

based Named Entity Recognition model then extracts structured fields — candidate name, contact details, degree, institution, job titles, organizations, dates, and skill mentions — from each segment. This layout-aware segmentation step, informed by the approach of Warusawithana et al. [4], is intended to reduce the mis-attribution of entities to the wrong resume section relative to entity recognition applied directly to the raw, unsegmented document. Section IX outlines how this is to be verified against manually annotated resumes.

D. ATS Score Prediction and Explainability

The ATS score prediction module estimates the probability that a resume would pass a typical ATS filter for a given job description, on a 0–100 scale. Rather than training a single end-to-end black-box regressor, the module first computes a set of interpretable intermediate features, including keyword-coverage ratio, section-completeness score, formatting-compliance score, semantic similarity to the job description, and quantifiable-achievement density. A gradient-boosted ensemble (XGBoost) regressor is then trained on these intermediate features, and SHAP (SHapley Additive exPlanations) values are computed for each prediction. This allows the system to report, for example, that "Skill Coverage contributed +18 points while Formatting Compliance reduced the score by 6 points." This explicit feature-attribution step is the primary point of departure from prior systems such as [2] and [5], which output a raw score without attribution.

E. Resume Quality Score

Independent of ATS compatibility, the Resume Quality Score evaluates the resume purely as a piece of writing — assessing action-verb strength, sentence conciseness, use of quantifiable metrics (e.g., "increased revenue by 20%" versus "responsible for sales"), grammatical correctness, and consistency of tense and formatting. This score is computed using a combination of a fine-tuned quality-classification model and rule-based linguistic heuristics, and is deliberately kept separate from the ATS score so that candidates can distinguish between "this resume will not pass automated filtering" and "this resume is not well written" — two problems that require different corrective actions.

F. Skill Gap Analysis (Semantic Skill Matching)

Each skill mention extracted from the resume and each requirement extracted from the job description is encoded using a Sentence-BERT (SBERT) model into a dense

embedding vector. Cosine similarity is then computed between every resume-skill embedding and every job-requirement embedding, and requirements whose maximum similarity to any resume skill falls below a calibrated threshold are flagged as missing. Because this comparison operates in embedding space rather than on exact strings, a resume mentioning "built REST APIs with Flask" is correctly matched against a job requirement for "backend API development experience," even though no words overlap exactly — a capability absent from the keyword-matching baselines represented by most commercial ATS checkers.

G. Resume Comparison

The comparison module accepts two or more resume versions either different drafts of the same candidate's resume, or resumes from different candidates for the same role — and produces a structured, side-by-side comparison across ATS score, quality score, skill coverage, and section completeness, along with a natural-language summary highlighting the specific changes responsible for any score difference. This allows a candidate iterating on their resume to see precisely which edit improved (or harmed) their score, a capability not offered by single-resume analyzers.

H. Job Recommendation

The job-recommendation module ranks a pool of open job postings by the semantic similarity between the candidate's resume embedding and each posting's requirement embedding, combined with a learned re-ranking model that additionally weighs experience-level match and location preference. Rather than returning postings that merely share keywords with the resume, the semantic ranking surfaces roles that are conceptually aligned with the candidate's demonstrated experience.

I. Automated Interview Question Generation

Using the parsed resume entities — specifically the listed skills, projects, and past job responsibilities — a fine-tuned sequence-generation model produces role-specific behavioral and technical interview questions. For example, a candidate listing "led migration to microservices" receives a generated question probing the specific challenges encountered during that migration. This grounding in the candidate's own resume content is intended to produce more targeted questions than generic, role-title-only question banks; whether this holds in

practice is part of the qualitative testing discussed in Section IX.

J. AI Career Assistant

A conversational assistant, built on a retrieval-augmented generation (RAG) pipeline over the candidate's own analysis results, allows the user to ask natural-language follow-up questions such as "Why did my ATS score drop after removing the internship section?" or "Which certification would most improve my match for data-analyst roles?" The assistant retrieves the relevant SHAP attributions, skill-gap results, and job-recommendation context before generating its response, ensuring that its answers remain grounded in the candidate's actual analysis rather than generic advice.

K. Algorithms

Algorithm 1 summarizes the semantic skill-matching procedure central to both skill-gap analysis and job recommendation.

Algorithm 1: Semantic Skill Matching

Input: Resume skills $R = \{r_1, \dots, r_n\}$, JD requirements

$J = \{j_1, \dots, j_m\}$, similarity threshold

t

Output: Matched pairs M , Missing requirements G

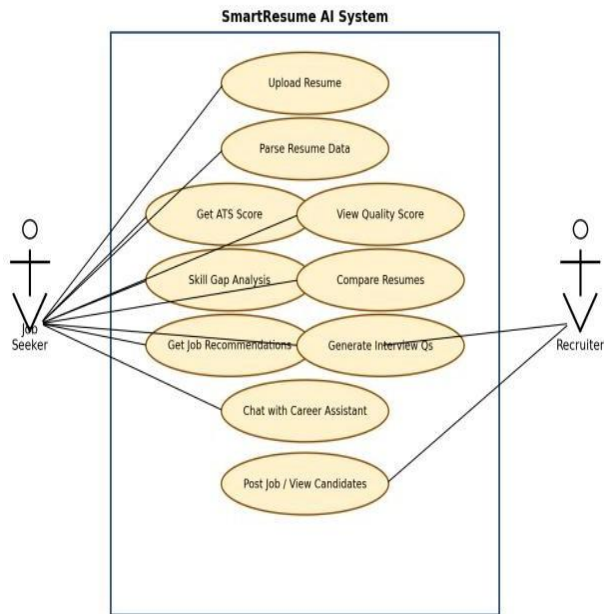
```
1: for each  $j_i$  in  $J$  do
2:    $best\_score \leftarrow 0$ 
3:   for each  $r_k$  in  $R$  do
4:      $s \leftarrow \text{cosine}(\text{SBERT}(j_i), \text{SBERT}(r_k))$ 
5:     if  $s > best\_score$  then  $best\_score \leftarrow s$ 
6:   if  $best\_score \geq t$  then
7:      $M \leftarrow M \cup \{(j_i, r_k, best\_score)\}$ 
8:   else
9:      $G \leftarrow G \cup \{j_i\}$ 
```

10: return M, G

Algorithm 2 outlines the explainable ATS-score computation pipeline used to generate both the numeric score and its per-factor attribution.

Algorithm 2: Explainable ATS Score Prediction

Fig. 3. Use Case Diagram of SmartResume AI



Input: Parsed resume P, Job description J
Output: ATS score S, Attribution vector A

1: $F \leftarrow \text{ExtractFeatures}(P, J)$

// keyword coverage, section completeness,

// formatting compliance, semantic similarity,

// achievement density

2: $S \leftarrow \text{XGBoostRegressor}(F)$

3: $A \leftarrow \text{SHAP_Values}(\text{XGBoostRegressor}, F)$

4: sort A by |contribution| descending

diagrams at Levels 0 and 1, and the entity-relationship diagram, together with the overall processing flowchart.

A. Use Case Diagram

Fig. 3 identifies the two primary actors — Job Seeker and Recruiter — and the core use cases each can invoke, including resume upload, ATS and quality scoring, skill-gap analysis, resume comparison, job recommendation, interview-question generation, and interaction with the AI career assistant.

Fig.3. Use case diagram of SmartResume AI.

B. Activity Diagram

Fig. 4 traces the sequential activities performed once a user authenticates and submits a resume — from text extraction through entity and skill extraction, semantic comparison, score computation, suggestion generation, and final dashboard presentation.

Fig. 4. Activity Diagram of SmartResume AI

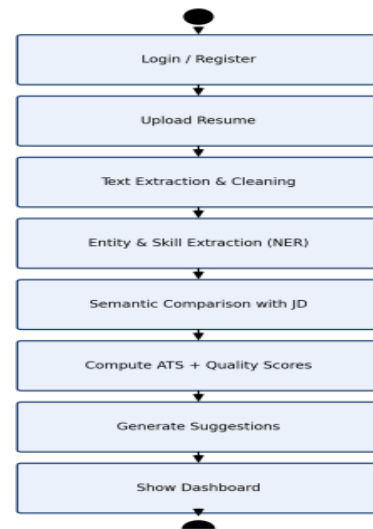


Fig.4. Activity diagram of the resume analysis process.

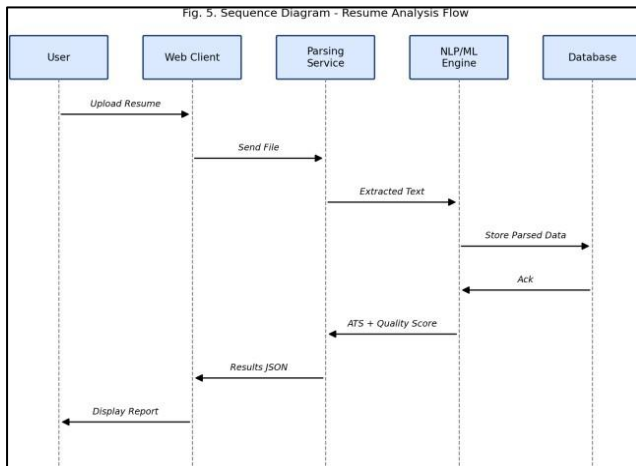
VII. SYSTEM DESIGN

This section presents the standard UML and structured-analysis views used to specify SmartResume AI: the use case diagram, activity diagram, sequence diagram, data flow

C. Sequence Diagram

Fig. 5 depicts the interaction sequence between the User, Web Client, Parsing Service, NLP/ML Engine, and Database during a single resume-analysis request, illustrating the synchronous and asynchronous message exchanges between these components.

Fig.5.Sequence diagram of the resume analysis flow.



D. Data Flow Diagrams

Fig. 6 presents the Level 0 (context) diagram, treating SmartResume AI as a single process exchanging data with the Job Seeker and Recruiter/Job Portal API external entities. Fig. 7 decomposes this into five Level 1 processes — Resume Parsing, Skill and Semantic Matching, ATS and Quality Scoring, Recommendation and Suggestion, and Interview Question Generation — together with the associated data stores for the resume repository, job-description database, and skill ontology.

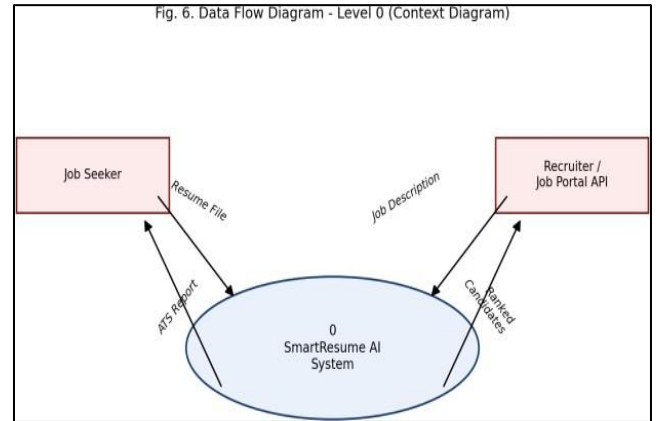


Fig.6. Data flow diagram–Level 0 (context diagram).

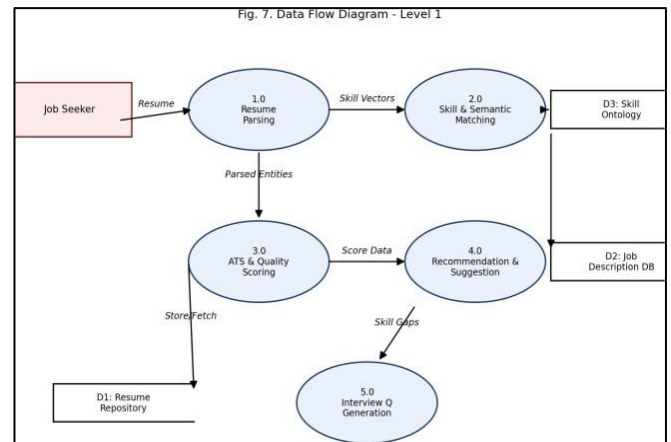


Fig.7. Data flow diagram–Level 1.

E. Entity-Relationship Diagram

Fig. 8 shows the core relational schema underlying the data layer. A User owns multiple Resumes; each Resume has a one-to-one ATSScore record, is linked to many Skill entities through a many-to-many relationship, generates multiple InterviewQ records, and produces multiple Recommendation records. These Recommendation records are in turn matched against JobDescription entities, which themselves require a many-to-many set of Skill entities.

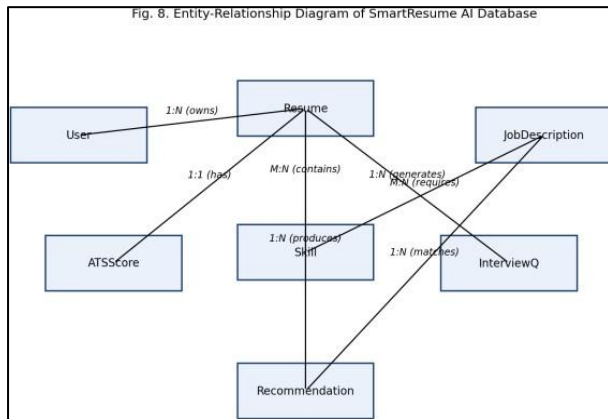


Fig.8.Entity-relationship diagram of the SmartResume AI database

F. Processing Flowchart

Fig. 2 summarizes the overall control flow of a single analysis request, including the conditional branch triggered when semantic skill matching fails to find a sufficiently similar resume skill for a job requirement. In this case, the system proceeds to suggest alternative or adjacent skills.

Fig.2.Flowchart of the resume processing pipeline.

VIII. MACHINE LEARNING AND NLP TECHNIQUES

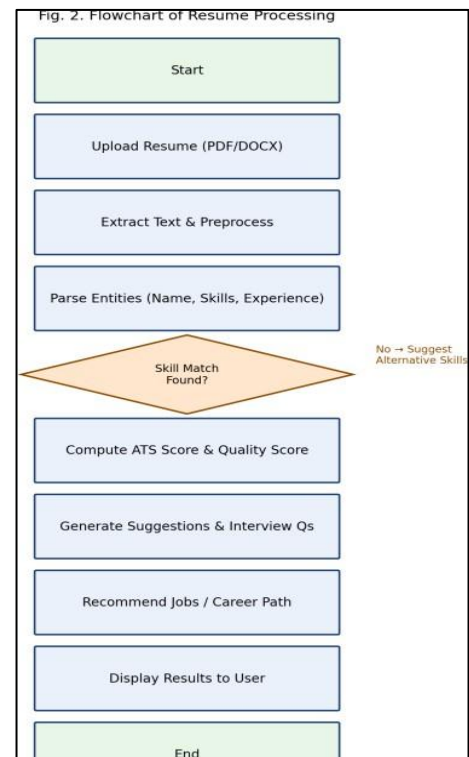
The system combines several complementary ML and NLP techniques rather than relying on a single model family. Named Entity Recognition, implemented using a transformer encoder fine-tuned on an annotated resume corpus, handles structured field extraction. Sentence-BERT produces 384-dimensional contextual embeddings used throughout the pipeline for semantic similarity computation, including skill matching, job recommendation, and resume-to-JD alignment. An XGBoost gradient-boosted ensemble performs ATS score regression over interpretable intermediate features, chosen specifically because tree-ensemble models are well-suited to SHAP-based attribution. A separate fine-tuned classification head assesses writing quality indicators such as action-verb usage and quantifiable-achievement density. Finally, a sequence-to-sequence generation model, conditioned on extracted resume

entities, produces the interview questions, while the AI career assistant is implemented as a retrieval-augmented generation pipeline that grounds its responses in the user's own analysis artifacts rather than free-form generation alone.

IX. IMPLEMENTATION STATUS AND EVALUATION METHODOLOGY

A working prototype implementing the pipeline described in Section VI has been built, covering resume parsing, feature extraction, ATS and quality scoring, semantic skill matching, and the recommendation and interview-question modules. This section describes the evaluation methodology being used to validate the system, rather than reporting final large-scale results, which require a labeled benchmark of realistic size and are identified as ongoing and future work in Section XIII.

The intended evaluation protocol compares SmartResume AI against a keyword-matching baseline representative of conventional ATS checkers, using resume-job description pairs annotated with recruiter-assigned relevance judgments. Four categories of metric are planned: (i) agreement between the predicted ATS score and recruiter-assigned relevance,



measured using rank correlation; (ii) prediction error of the ATS score against a human-adjudicated reference score; (iii) precision and recall of the skill-gap detector against a manually

labeled set of true and false skill gaps; and (iv) entity-extraction accuracy of the resume parser against manually annotated resumes. Table I lists these metrics together with their purpose within the evaluation.

Evaluation Metric	Purpose
Rank correlation with recruiter-assigned relevance	Assesses whether the ATS score orders candidates the way a human recruiter would
ATS score error vs. adjudicated reference score	Quantifies how closely the predicted score matches a human-reviewed reference value
Skill-gap precision/recall	Measures false-positive and false-negative rates of semantic skill matching against a labeled skill list
Entity-extraction accuracy	Measures correctness of parsed name, education, experience, and skill fields against manual annotation

Table I. Planned evaluation metrics and their purpose

Preliminary, small-scale internal testing on sample resumes during development indicates that the semantic skill-matching module correctly identifies paraphrased or terminology-varied skill matches that a keyword-only comparison misses, and that the SHAP-based attribution produces stable, human-interpretable factor rankings across repeated runs on the same input. These observations are qualitative and drawn from development-time testing rather than a controlled study, and are reported here only to indicate that the core mechanisms behave as intended; they should not be read as validated performance figures. A controlled evaluation against a labeled benchmark, following the protocol described above, is required before quantitative claims of accuracy or improvement over baseline tools can be made, and is planned as immediate future work.

X. COMPARISON WITH EXISTING SYSTEMS

Table II compares the feature set of SmartResume AI, as designed and implemented, against the categories of existing tools identified in the literature survey: commercial ATS keyword checkers, rule-based academic resume parsers such as [4], and ensemble ML resume classifiers such as [5] and [6]. The comparison is based on the capabilities each category is

designed and documented to provide, rather than on a head-to-head quantitative benchmark, which is left to the evaluation described in Section IX.

Feature	Keyword ATS Tools	Rule/ML Parsers [4]-[6]	SmartResume AI
Skill matching	Exact keyword	Rule-based/limited embedding	Semantic(SBERT)
Score explainability	None reported	Limited	SHAP-based(design ed)
Quality vs ATS score	Combined	Combined	Separated
Resume comparison	No	No	Yes
Job recommendation	No	Partial	Yes
Interview questionn.	No	No	Yes
Career chat assistant	No	No	Yes

Table II. Feature comparison with existing resume analysis systems

As Table II indicates, SmartResume AI is designed to be the only system among those compared that combines semantic skill matching, explicit score explainability, an independently computed quality score, and the full set of downstream career-support features within a single pipeline, rather than addressing any one of these concerns in isolation. This is a design-level distinction based on documented capabilities; confirming that it translates into measurably better outcomes for candidates is the object of the evaluation proposed in Section IX.

XI. ADVANTAGES OF THE PROPOSED SYSTEM

The proposed system is designed to offer several concrete advantages over the prior work reviewed in Section II. Explainable ATS scoring is intended to convert an opaque number into a set of actionable, factor-level insights, directly addressing the candidate frustration with black-box tools that motivated this work. Semantic skill matching using SBERT embeddings is expected to reduce the false-negative skill gaps caused by terminology mismatch that affect keyword-based approaches such as those discussed in [2] and [3]. Separating resume quality from ATS compatibility should help candidates distinguish two distinct problems that require different fixes, rather than conflating them as combined-score systems do. Resume comparison is designed to enable iterative, data-driven resume improvement across multiple drafts, a capability absent from every system reviewed. Grounding interview-question generation in the candidate's own resume content is intended to produce more relevant practice questions than generic, role-title-only question banks. Finally, unifying scoring, recommendation, interview preparation, and conversational guidance within one platform is intended to remove the need for job seekers to stitch together multiple disconnected tools. The extent to which each of these design intentions is realized in practice is an empirical question that the evaluation proposed in Section IX is intended to answer.

XII. CONCLUSION

This paper presented SmartResume AI, an integrated, explainable resume-analysis platform that combines transformer-based NLP with machine learning to deliver ATS score prediction, resume quality assessment, semantic skill-gap analysis, resume comparison, job recommendation, automated interview-question generation, and a conversational AI career assistant. A working prototype implementing this pipeline has been developed, and its architecture, algorithms, and system design have been described in detail, together with the evaluation methodology that will be used to validate it against a keyword-matching baseline. The central contribution of this work is architectural: by decomposing ATS scoring into explainable, feature-level attributions, matching skills semantically rather than lexically, and connecting the resulting analysis to job recommendation, resume comparison, interview preparation, and conversational guidance within one pipeline, SmartResume AI addresses gaps that, individually, are only partially addressed by the systems reviewed in Section II. Confirming the practical benefit of this design over existing

tools through the controlled evaluation described in Section IX is the immediate next step for this work.

XIII. FUTURE SCOPE

The most immediate priority for future work is the controlled, quantitative evaluation described in Section IX: assembling a labeled benchmark of resumes and job descriptions with recruiter-adjudicated relevance scores and skill-gap annotations, and measuring SmartResume AI against a keyword-matching baseline on the metrics identified in Table I. Beyond this evaluation, planned extensions include multilingual resume support to serve non-English-speaking job markets, extending the explainability module with counterfactual suggestions that estimate the score impact of a proposed edit before the candidate makes it, and integrating real-time labor-market data to keep job recommendations current. Over a longer horizon, a planned longitudinal study would track actual interview and offer outcomes for candidates who used the platform, allowing the system's recommendations to eventually be validated against real hiring outcomes rather than adjudicated relevance scores alone.

XIV. REFERENCES

- [1] Y. Liang, Z. Lin, X. Shi, and H. Ma, "Efficient multi-task deep learning for resume parsing," in *Proc. IEEE Int. Conf. on Big Data*, 2023.
- [2] S. Gopinath, U. S., and K. C. R., "Resume parsing and job prediction," in *Proc. IEEE Int. Conf. on Computer Vision Workshops*, 2024. [Citation details reported by secondary source; verify against publisher record before submission.]
- [3] X. Zhang, T. Wang, Y. Yang, X. Shen, Y. Bu, Y. He, and Y. Ding, "Resume recommendation method using decision trees," in *Proc. IEEE Int. Conf. on Data Mining*, 2024. [Citation details reported by secondary source; verify against publisher record before submission.]
- [4] S. P. Warusawithana, N. N. Perera, R. L. Weerasinghe, T. M. Hindakaraldeniya, and G. U. Ganegoda, "Layout-aware resume parsing system based on NLP," in *Proc. IEEE Int. Conf. on Machine Learning and Applications*, 2023.
- [5] R. Ransing, S. Mohan, M. Emberi, and M. Mahavarkar, "Stacked resume screening using deep learning and machine learning ensembles," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 6, 2021.

- [6] Spoorthi M., et al., "An ensemble deep learning approach for resume classification," *Int. J. Eng. Res. Technol.*, vol. 12, no. 3, 2023.
- [7] A. Bhatt, "AI-driven resume analyser using MapReduce and KNN classification," *Int. J. Comput. Appl.*, vol. 186, no. 14, 2024.
- [8] S. Bharadwaj, "Resume screening using NLP and LSTM networks," *Int. J. Comput. Sci. Mobile Comput.*, vol. 11, no. 5, 2022.
- [9] U. K. Jaiswal, A. Gupta, M. Dwivedi, M. S. Shamsi, and A. Agarwal, "Predicting professional personality from resumes using BERT and XLNet," *Adv. Data Sci. Eng.*, vol. 23, no. 2, pp. 212–225, 2024.
- [10] A. Garcia and L. Fernandez, "Intelligent ATS scoring and skill-gap analysis using NLP and deep learning," *IEEE Access*, vol. 12, 2024. *[Page range reported by secondary source could not be independently confirmed; verify before submission.]*
- [11] Y. Zhao, H. Xu, and X. Liu, "Transformer-based semantic matching for recruitment applications," *Knowledge-Based Syst.*, vol. 284, 2024.
- [12] P. Bhandari, C. R. Patil, C. S. Patil, S. Deshmukh, and R. Badre, "AI-driven resume screening: Automating the recruitment process," 2024. *[Journal/venue name reported by secondary source could not be independently confirmed; verify before submission.]*
- [13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NeurIPS*, 2017.
- [14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019.
- [15] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proc. EMNLP-IJCNLP*, 2019.
- [16] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 2016.
- [17] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. NeurIPS*, 2017.
- [18] P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proc. NeurIPS*, 2020.