ResumeMatch: A Cloud-Native SaaS Platform for Semantic Resume-Job Alignment

Kabir Roy
Department of Computer Science & Engineering
Affiliation/University Name
City, Country
email: kabir.roy@example.com

***Abstract*—We present ResumeMatch, an open-source cloud-native SaaS application that applies established Natural Language Processing techniques to the resume screening problem. Unlike traditional Applicant Tracking Systems that rely on rigid keyword matching, ResumeMatch implements a hybrid comparison engine utilizing Sentence-BERT for semantic similarity and explicit skill extraction for interpretability. This paper describes the system architecture and includes a rigorous evaluation on a gold-standard dataset of 100 resume-job pairs. Results demonstrate statistically significant improvements ($p<0.001$) over TF-IDF baselines, confirming the platform's viability as a real-time tool for democratizing access to advanced NLP in career technology.**

Keywords—Applied NLP, System Design, Sentence Transformers, Recruitment Technology, Open Source

I. INTRODUCTION

The mismatch between candidate qualifications and job description phrasing represents a persistent challenge in automated recruitment. While transformer-based matching has been extensively studied [1, 2], few accessible, production-ready tools implement these techniques in a user-facing format suitable for individual job seekers.

ResumeMatch addresses this implementation gap by providing an open-source, cloud-deployable platform that makes State-of-the-Art NLP accessible beyond enterprise Applicant Tracking Systems. Our contribution is not a novel algorithm, but rather a thoughtful system integration that balances semantic understanding with interpretability, wrapped in a production-ready architecture.

A. Design Goals

Semantic Matching: Move beyond keyword overlap to contextual understanding.
Explainability: Provide transparent scoring with identified skill gaps.
Actionability: Generate concrete improvement suggestions via LLMs.
Accessibility: Free, open-source tool for job seekers.
Performance: Real-time response suitable for interactive use.

II. SYSTEM ARCHITECTURE

ResumeMatch is built as a decoupled microservices architecture to ensure scalability, maintainability, and cloud portability.

A. Technology Stack

1) *Frontend Layer*: React.js with Vite bundler for optimal load performance. Custom CSS with neon-themed UI for visual distinction. Responsive design for mobile and desktop.
2) *Backend Layer*: FastAPI (Python 3.10+) for high-performance async request handling. Uvicorn ASGI server with WebSocket support for streaming. RESTful API design with OpenAPI documentation.
3) *NLP Pipeline*: sentence-transformers library for SBERT inference. Model: all-MiniLM-L6-v2 (80MB, 384-dimensional embeddings). pdfminer.six for PDF text extraction with layout awareness.
4) *Generative Layer*: Groq API integration for Llama-3-8B-Instant inference. Structured JSON output prompting for resume recommendations. Fallback to local generation if API unavailable.
5) *Deployment*: Vercel (edge deployment). Backend: Render (containerized Python service). GitHub Actions CI/CD for automated testing.

B. Processing Pipeline

The system processes resume-JD pairs through a four-stage pipeline:
Stage 1: Document Parsing. Input: PDF Resume yields Plain Text via pdfminer.six. Preserves formatting for contact extraction.
Stage 2: Skill Extraction. Explicit Skills: Regex patterns for technologies. Output: Set of detected skills with frequency counts.
Stage 3: Semantic Embedding. SBERT Encoding: Resume Text and JD Text mapped to 384-dim vectors. Cosine Similarity.
Stage 4: Hybrid Scoring. S_final = alpha * S_skills + beta * S_semantic. Where S_skills = Jaccard index, S_semantic = cosine similarity.

C. Generative Feedback Engine

When a match score is below threshold or upon user request, the system invokes the Llama-3 model via Groq's inference API with a structured prompt. The structured output ensures reliability and allows direct UI rendering without additional parsing.

III. IMPLEMENTATION DETAILS

A. Performance Optimization

Cold Start Mitigation: SBERT model loaded once at server initialization.
Inference Optimization: Batch processing for multiple resume comparisons. Cached embeddings for frequently-used JD templates.
Resource Management: Model runs on CPU (80MB RAM footprint). Suitable for free-tier cloud hosting (512MB instances).

B. Latency Benchmarks

Testing environment: Render Free Tier (512MB RAM, shared CPU).

Operation	Latency
PDF Parsing	45-120ms
SBERT Inference	12-18ms
Skill Extraction	3-8ms
Hybrid Scoring	<1ms
Groq API Call	800-1500ms
Total (Interactive)	~180ms

Table I. Latency Benchmarks

These measurements demonstrate that the SBERT architecture adds minimal overhead (<20ms) while providing significant semantic capability.

C. User Experience Design

Progressive Disclosure: Immediate visual feedback during upload. Parsed resume preview with detected information.
Explainability Features: Color-coded skill badges (matched vs. missing vs. bonus). Visual match score gauge with percentage.

IV. QUANTITATIVE EVALUATION

To validate the system's accuracy beyond theoretical benefits, we conducted a rigorous comparative analysis against a standard TF-IDF baseline.

A. Experimental Setup

To quantitatively validate semantic matching quality, we constructed a gold-standard dataset of 100 resume–job description pairs spanning 10 technical roles and 3 experience levels (junior, mid-level, senior). Each pair was manually labeled across 5 match quality levels: High, Medium-High, Medium, Low-Medium, and Low, based on skill alignment, experience relevance, and domain fit.

The dataset ensures diversity across technical domains (ML, web development, DevOps) and match quality, providing robust evaluation coverage across realistic hiring scenarios.

B. Cross-Validation Protocol

To assess generalization and avoid overfitting, we employed 5-fold cross-validation on the 100-pair dataset. Each fold contains 80 training pairs and 20 test pairs, with stratified sampling to ensure balanced representation of match quality levels. Table I reports mean Spearman correlation and MSE metrics.

C. Accuracy Benchmarks

We evaluated three models: Baseline TF-IDF, ResumeMatch (SBERT), and Hybrid. Table II presents the comparative results with 95% confidence intervals derived from 5-fold cross-validation.

Model	**Spearman ρ (95% CI)**	**MSE (95% CI)**	**p-value**
TF-IDF	0.74 (0.71-0.77)	0.22 (0.19-0.25)	-
SBERT	0.84 (0.82-0.86)	0.04 (0.03-0.05)	<0.001
Hybrid	0.85 (0.83-0.87)	0.03 (0.02-0.04)	<0.001

Table II. Comparitive Accuracy (n=100)

The consistently higher correlation confirms transformer superiority in capturing non-keyword relevance (p<0.001). SBERT scores align significantly closer to human intuition (0.04 MSE vs 0.22 for TF-IDF).

C. Parameter Optimization

We performed a Grid Search to determine optimal weights for alpha (Skill) and beta (Semantic). The optimization converged at beta approx 1.0, indicating that Semantic Vector Similarity is the dominant predictor of fit.

V. DISCUSSION

A. System Contributions

1) *Democratized NLP Access*: ResumeMatch brings transformer-based semantic matching to individual job seekers.
2) *Hybrid Interpretability*: Combining explicit skill extraction with semantic similarity addresses the 'black

box' criticism.
3) *Constructive Feedback Loop*: Paradigm shift from 'screening' to 'coaching'.

B. Limitations and Future Work

Current limitations include regex-based skill extraction (English-centric) and standard resume formats. Planned improvements include fine-tuning SBERT on domain-specific data and adding experience level weighting.

C. Ethical Considerations

Privacy is prioritized with server-side processing and no persistent storage. Open-source nature allows scrutiny of algorithms to prevent proprietary gatekeeping.

VI. RELATED WORK

Resume-Job Matching: Maheshwari et al. [1] demonstrated BERT's effectiveness. We extend this with hybrid scoring and generative feedback.

Sentence Transformers: Reimers & Gurevych [2] introduced SBERT. We apply this specifically to career documents.
Explainable AI in HR: Recent work [3, 4] emphasizes interpretability. Our hybrid approach directly addresses this.

VII. CONCLUSION

We presented ResumeMatch, a production-ready SaaS platform demonstrating established NLP in career technology. By integrating SBERT, regex, and GenAI, we provide a transparent, actionable tool for job seekers. Code is available at github.com/Kabirroy12345/resume-match-engine.

REFERENCES

[1] S. Maheshwari, S. Sajnani, and A. Garg, "Resume Screening using Bidirectional Encoder Representations from Transformers (BERT)," 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), 2020.

[2] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2019.

[3] J. Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," NAACL-HLT, 2019.

[4] A. Raghavan, S. Barocas, K. Levy, and S. Narayanan, "Mitigating Bias in Algorithmic Hiring: Evaluating Claims and Practices," Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, 2020.

[5] A. Vaswani et al., "Attention Is All You Need," Advances in Neural Information Processing Systems 30 (NIPS), 2017.