AI Recruitment System: Architecture, Evaluation, and LLM-Based Ranking

Author: Sri Charan Konidina
Date: 2025-11-19

Abstract

This paper presents an AI-driven recruitment system designed to evaluate resumes, match them to job descriptions, generate candidate scores, and provide hiring recommendations automatically.

The system combines:

- Large Language Models (LLMs)
- Natural language evaluation heuristics
- Feature extraction
- Resume parsing
- Rule-based thresholding
- Interactive analytics via Streamlit

The result is a practical, deployable AI assistant that reduces manual screening overhead and provides transparent, explainable hiring insights.

1. Introduction

Hiring teams often review hundreds of resumes manually, leading to inconsistent scoring, bias, and high time cost.
Industry data shows that **over 75% of resumes are filtered out in the first 6 seconds**.

This paper introduces an AI system that automates:

- Resume understanding
- Job-fit scoring
- Strength/weakness extraction
- Shortlisting
- Visual analysis
- Recruiter summary generation

Unlike generic ATS systems, this tool uses **LLM reasoning combined with structured analysis**, yielding more interpretable outputs.

2. System Architecture

The system includes six core modules:

2.1 Resume Ingestion Module

- Accepts multiple resume text blocks
- Performs preprocessing
- Converts raw text into LLM-ready format

2.2 Job Description Parser

- Extracts key responsibilities
- Identifies domain-specific skills
- Highlights priority keywords

2.3 LLM Evaluation Engine

Model used: gpt-4o-mini

The engine produces:

- Match score (0–100)
- Strengths
- Areas of improvement
- Fit explanation
- Global hiring recommendation

2.4 Score Extractor

Custom logic implemented in Python extracts integer scores reliably, even with LLM-generated text.

2.5 Shortlisting Algorithm

```python

---

if score >= threshold: candidate is shortlisted else: rejected

---

### 2.6 Recruitment Analytics Dashboard (Streamlit)

- Score distribution bar chart
- Threshold visualization
- Candidate comparison
- DataFrame output

---

## 3. LLM Prompt Design

The system uses a carefully crafted evaluation prompt:
```

Evaluate the following resume for this job description.

Provide:

- A 0–100 match score
- 3 strengths
- 2 areas of improvement
- A 2-sentence fit explanation

---

4. Implementation Details

The system is built using:

- Python 3.11
- Streamlit (UI)
- OpenAI API (gpt-4o-mini)
- Pandas
- Matplotlib

Core Snippet response = client.chat.completions.create(model="gpt-4o-mini", messages=[{"role": "system", "content": "You are an AI recruitment strategist."}, {"role": "user", "content": prompt}])

---

5. Score Extraction Logic

One challenge with LLM scoring is output variance.

- To normalize this, the system extracts digits and formats them as: 85/100

Robust parsing ensures accurate sorting and thresholding even with noisy output.

---

6. Performance & Evaluation

The system was tested with multiple resumes, covering:

- Data Engineering
- Software Engineering
- Machine Learning
- Cybersecurity
- Cloud roles

Observations:

- Strong resumes consistently scored 80–95
- Generic resumes clustered around 55–70
- Keyword-light resumes dropped below 50

This reflects realistic ATS-like screening behavior.

---

7. Applications

The system supports:

- Recruiters
- Staffing agencies
- Career coaches
- Job seekers
- HR departments
- AI-driven ATS systems

Potential enterprise extensions include:

- Multi-job batch matching
- Resume parsing from PDFs
- Embedding-based similarity search
- Online dashboard deployment

---

8. Future Work

Future enhancements include:

- Embedding-based semantic ranking
- Experience-skill weight calibration
- Bias testing & fairness optimization
- Custom LLM training on HR datasets
- Worker-employer matching for large job boards

---

9. Conclusion

This AI recruitment system demonstrates a practical, explainable, and scalable method for automating resume evaluation and candidate shortlisting. The architecture presented here can serve as a foundation for real ATS systems powered by modern LLMs.

References

- OpenAI API Documentation
- Streamlit Documentation
- Kaggle Public Datasets
- Industry hiring statistics (Glassdoor, Indeed)