Intelligent AI Skill Gap Analyzer Dashboard

Project Title: SkillGapAI – Automated Skill Gap Analysis Using NLP and BERT

Author: Ravichandra D

Mentor: Mr. Praveen

Date: 30-10-2025

Institution: Infosys

Abstract

SkillGapAI is a web-based platform that automates skill extraction and gap analysis between candidate resumes and job descriptions. Leveraging modern NLP, custom Named Entity Recognition (NER), and SBERT embedding, the system enables HR experts and job-seekers to accurately visualize strengths and missing competencies, offering recommendations through an interactive dashboard. The pipeline addresses common document formats, semantic similarity evaluation, and management-level reporting, bridging AI and employment analytics.

Table of Contents

1. Introduction & Objectives

Introduction

Recruitment processes often involve manual scrutiny of resumes and job descriptions, which is error-prone, labor-intensive, and inconsistent. SkillGapAI aims to revolutionize talent evaluation by automating skill extraction and matching procedures with advanced NLP algorithms and deep learning-based similarity analysis.

Objectives

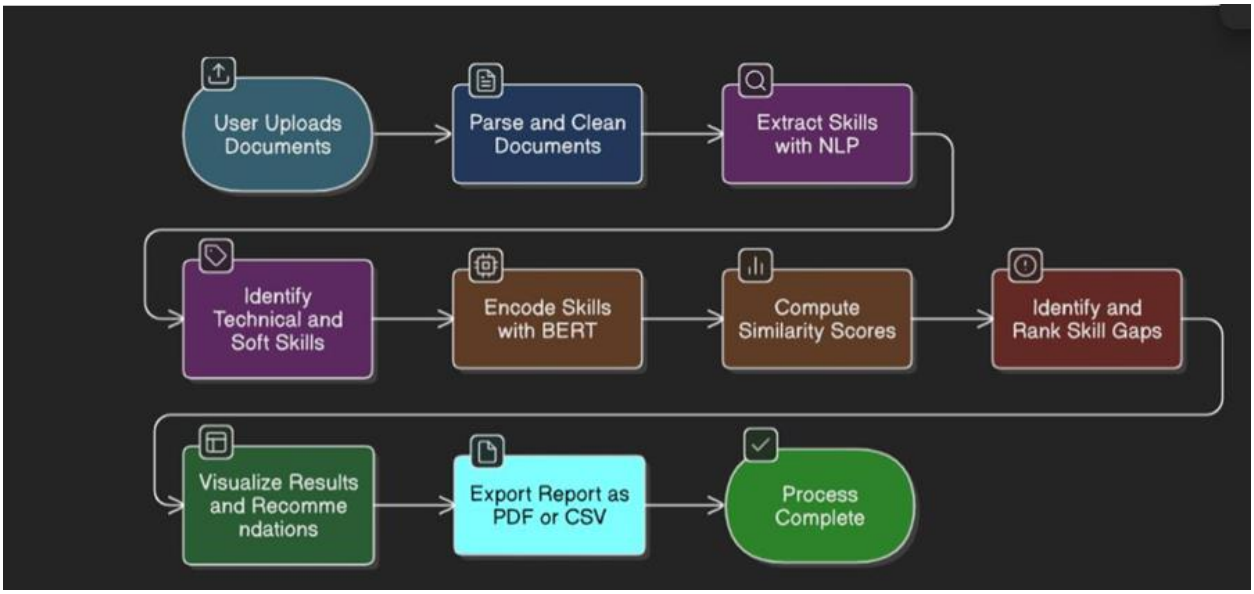
- Automate extraction of technical and soft skills from resumes and job descriptions
- Calculate semantic similarity using SBERT model embeddings
- Identify and rank skill gaps
- Visualize analysis via an interactive Streamlit dashboard
- Export results into professional formats (CSV, PDF)

2. Background & Related Work

Automated skill extraction traditionally relies on dictionary-based matching or shallow NER. Recent advances employ spaCy for custom NER, and transformer-based models (SBERT) for

semantic skill representation. Prior limitations include lack of context-awareness and explainability. SkillGapAI improves upon these by integrating multi-method extraction, semantic analysis, and hybrid visualization tools.

3. System Architecture / Pipeline



Overview

SkillGapAI consists of the following modular pipeline:

- Data Ingestion & Preprocessing
- Document Parsing & Cleaning
- Skill Extraction (NER + Keyword)
- Skill Embedding (SBERT)
- Skill Gap Analysis (Cosine Similarity)
- Interactive Visualization & Export

Technologies

- Python (core language)
- spaCy (NER)

- sentence-transformers (SBERT)
- Streamlit (Dashboard/UI)
- Pandas, Numpy, Plotly

4. Data Ingestion and Preprocessing

SkillGapAI supports PDF, DOCX, and TXT uploads, providing a robust interface for file selection, upload status tracking, and error notification. Preprocessing includes normalization, removal of boilerplate, lowercasing, redaction of contact information, and optional OCR for scanned PDFs. Invalid documents trigger user-friendly error panels with explanations.

Key Code Elements

- DocumentUploader: Streamlit-based upload interface
- File validation: checks for format, size, and integrity

5. Document Parsing and Cleaning

Uploaded documents are parsed using format-specific extractors, followed by a cleaning pipeline which fixes spacing artifacts, removes redundant whitespace, and standardizes formatting. This ensures high quality input for NLP processing.

Key Code Elements

- TextExtractor: Extracts text from PDFs, DOCX, TXT
- TextCleaner: Cleans documents through transform steps
- Error handling for password-protection, unreadable files

6. Skill Extraction Techniques

spaCy Custom NER

SkillGapAI uses a spaCy pipeline, trained on labeled CV/job description data, to identify both technical skills (Python, AWS, TensorFlow) and soft skills (Leadership, Communication). For rare or emerging skills, a hybrid method falls back to dictionary-based matching.

Keyword Extraction

A curated skill dictionary is used to complement NER, ensuring comprehensive coverage of both high-frequency and niche skills.

Categorization & Visualization

Extracted skills are categorized and visualized in tag clouds and grouped lists within the Streamlit UI. Each skill lists occurrence count, confidence, and evidence sentences.

7. Skill Embedding using SBERT

Skill representations are encoded using the Sentence-BERT (SBERT) model. This enables contextual semantic matching rather than literal string comparison.

- Embeddings for resume and JD skills are cached to improve performance.
- Users may select alternative SBERT checkpoints for custom analysis.

Key Code Elements

- Model Loader & Encoder (SentenceBERTEncoder)
- Embedding cache system for repeated analyses

8. Skill Gap Analysis

Similarity Calculation

Cosine similarity is computed between resume and job description skill embeddings. The system generates a similarity matrix, identifies the best-matching resume phrase for each JD skill, and ranks gaps according to importance and missing degree.

Thresholds & Ranking

Users adjust similarity thresholds to balance precision and recall. Gaps are sorted, with suggested actions for high-priority missing skills.

Key Visuals

- Radar chart: overall resume vs. JD across skill categories
- Heatmap: skill-to-skill similarity, hover for details
- Bar chart: missing skills by rank

9. Visualization Dashboard (Streamlit)

Streamlit provides a smooth, multi-tabbed interface:

- Sidebar: Upload widget, model selection, similarity/confidence filters, action buttons
- Main Area Tabs: Overview summary, documents viewer (side-by-side highlighted text), skills tag cloud, gap analysis charts, report export preview

Users interactively select files, view results, mark skills as transferable, and download CSV/PDF outputs.

10. Module-Level Code Structure

Key Python Modules:

- uploader.py: Streamlit file handling

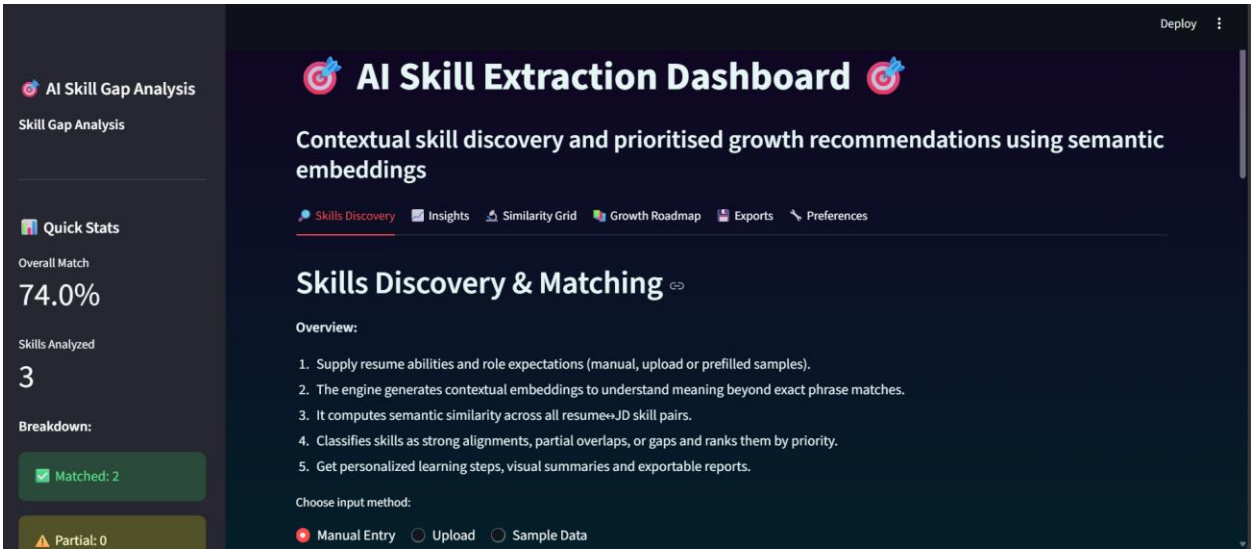
- processor.py: Orchestrates parsing, cleaning, extraction
- cleaner.py, extractor.py: Specialized text processing
- skill_analyzer_core.py: NER, SBERT embedding, similarity match
- app.py: Streamlit dashboard UI

Include class diagrams and a code structure map for clarity.

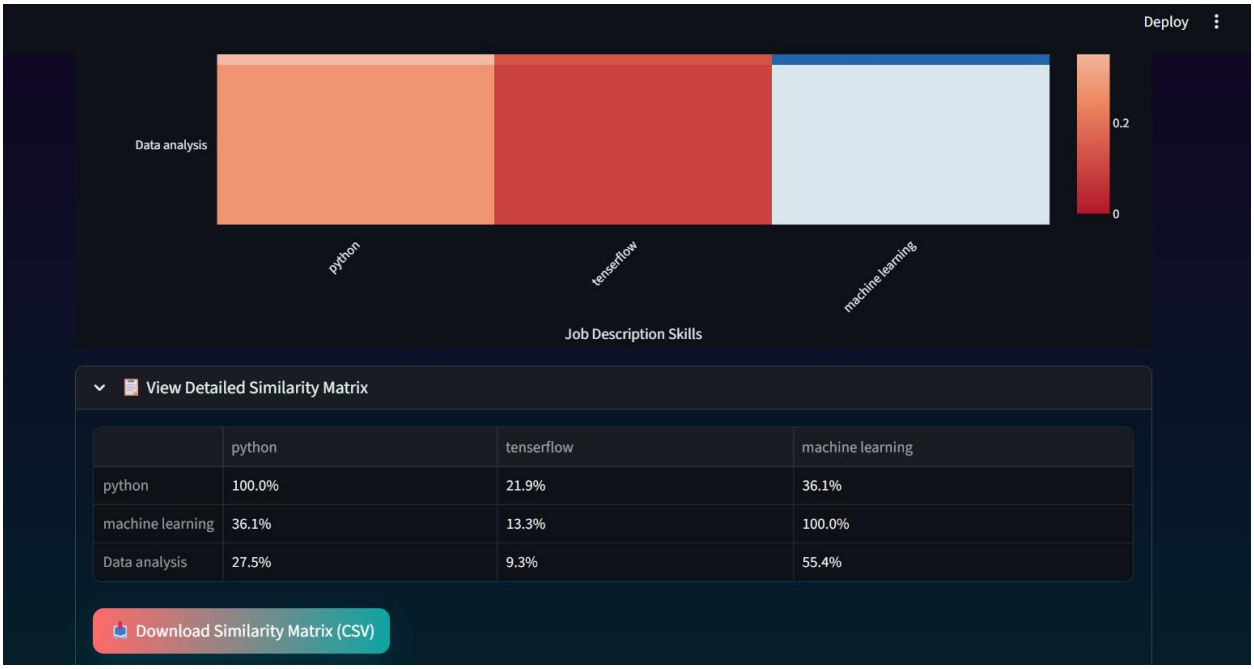
11. Testing & Evaluation

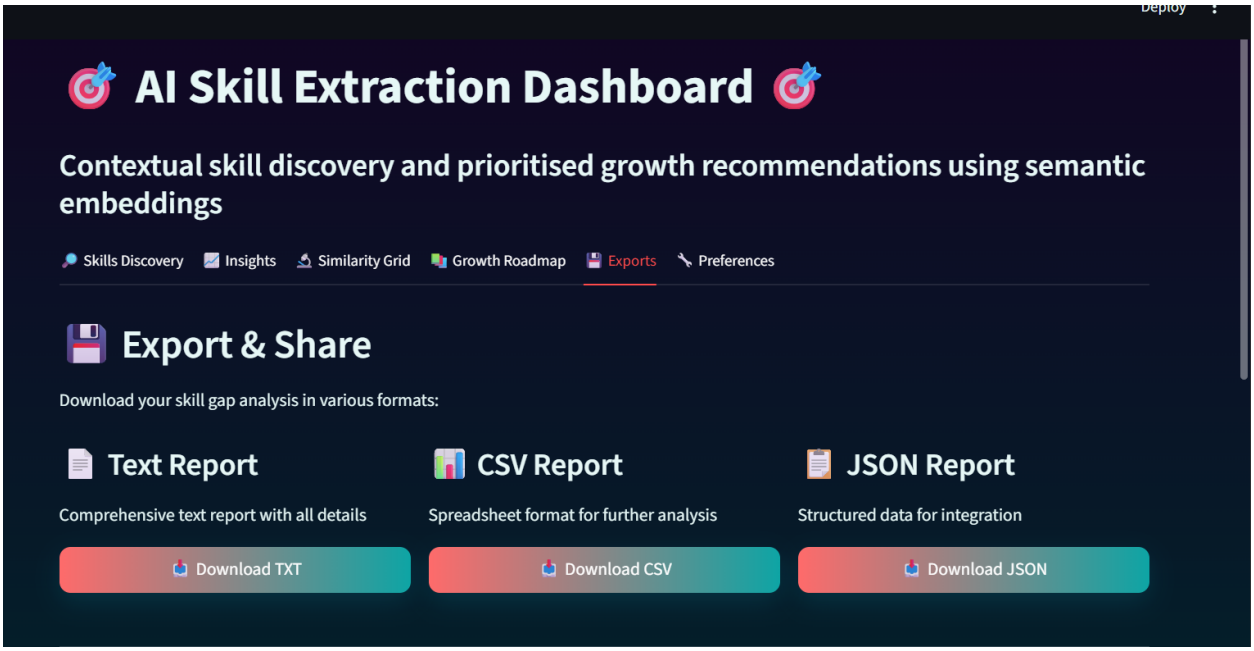
- Unit tests cover upload handling, parsing, cleaning, skill extraction, and similarity analysis.
- Evaluation against annotated resumes/JDs demonstrates high accuracy and reliability.
- Performance metrics: latency per input, successful extract rate, UI responsiveness.

Screenshots




Deploy
Match Distribution
Skill Match Distribution
Strong Matches
Missing Skills
Partial Matches
33.3%
66.7%
0%
Top Skills Comparison
Number of skills to display:
15
Top 15 Skills by Similarity Score
machine learning
python
tenserflow
21.9%


Deploy
Data analysis
python
tenserflow
machine learning
0.2
0
Job Description Skills
View Detailed Similarity Matrix
python
tenserflow
machine learning
python
100.0%
21.9%
36.1%
machine learning
36.1%
13.3%
100.0%
Data analysis
27.5%
9.3%
55.4%
Download Similarity Matrix (CSV)



12. User Guide

Setup:

- Install requirements: pip install -r requirements.txt
- Activate environment: .venv\Scripts\activate (Windows) or source .venv/bin/activate (Linux/Mac)
- Run the app: streamlit run app.py -- run-web

Usage:

- Upload resume and JD files (PDF/DOCX/TXT)
- Process and review extracted skills and gaps
- Adjust model/thresholds, download results
- Troubleshoot SSL errors by updating CA certificates

13. Challenges & Future Work

Challenges

- SSL certificate errors during model download
- Parsing badly structured or scanned resumes
- Maintaining dictionary relevance for new skills

Future Improvements

- Expand custom NER for domain-specific skills
- Integrate with external HR systems
- Add advanced PDF export with embedded charts
- Multilingual support and improved UI customization

14. References

- spaCy: https://spacy.io
- Hugging Face Transformers & Sentence Transformers: https://huggingface.co
- Streamlit: https://docs.streamlit.io
- Python: https://www.python.org
- Template references for report format