Resume Screener – Techniques Analysis

1. Summary of Research Papers

Paper	Goal	Techniques	Outcomes
Abstractive Text Summarization (Huawei)	Summarize resumes using NLP	BART, T5, Pegasus	BART-Large achieved highest ROUGE scores
Automated Personnel Selection (BRAC Univ.)	Match profiles to job criteria	RoBERTa, DistilBERT	RoBERTa: 85% accuracy
Automated Resume Parsing (Amrita Univ.)	Extract resume info	SpaCy NER, regex	Accurate extraction from varied formats
Competence-Level Prediction (Emory Univ.)	Match resumes to job descriptions	BERT + multi-head attention	73.3% accuracy
Resume Ranking (IRJET)	Rank resumes using keyword matching	TF-IDF, cosine similarity	Effective keyword-based ranking
Intelligent Resume Tracking (CMR College)	Optimize resumes	Gemini API	Real-time feedback and integration
Resume Matching Framework (Huawei Turkey)	OCR + NLP for resume matching	YOLOv8 (OCR), BERT, RoBERTa	High F1 scores (BERT: 0.93)
Resume Ranking (ICWITE 2024)	Cross-lingual ranking	DistilBERT, XLM	Efficient multilingual support

A. Keyword and Skill Matching – DistilBERT/XLM

DistilBERT is a lightweight version of BERT that balances accuracy and efficiency, making it suitable for fast keyword and skill extraction. It captures the context of words, enabling better understanding of job-related terms. **XLM** extends this capability to multilingual resumes, allowing effective processing of resumes in different languages. Compared to SpaCy NER and TF-IDF, DistilBERT and XLM provide better contextual understanding and higher accuracy.

B. Experience Quantification – BERT + Regex

BERT is a transformer model that understands complex sentence structures and context, making it ideal for extracting experience-related information. Regex complements BERT by identifying structured patterns like dates, job titles, and company names. This combination ensures accurate extraction from both structured and unstructured resumes. While SpaCy NER works well for structured data, BERT + Regex offers superior flexibility and accuracy for diverse formats.

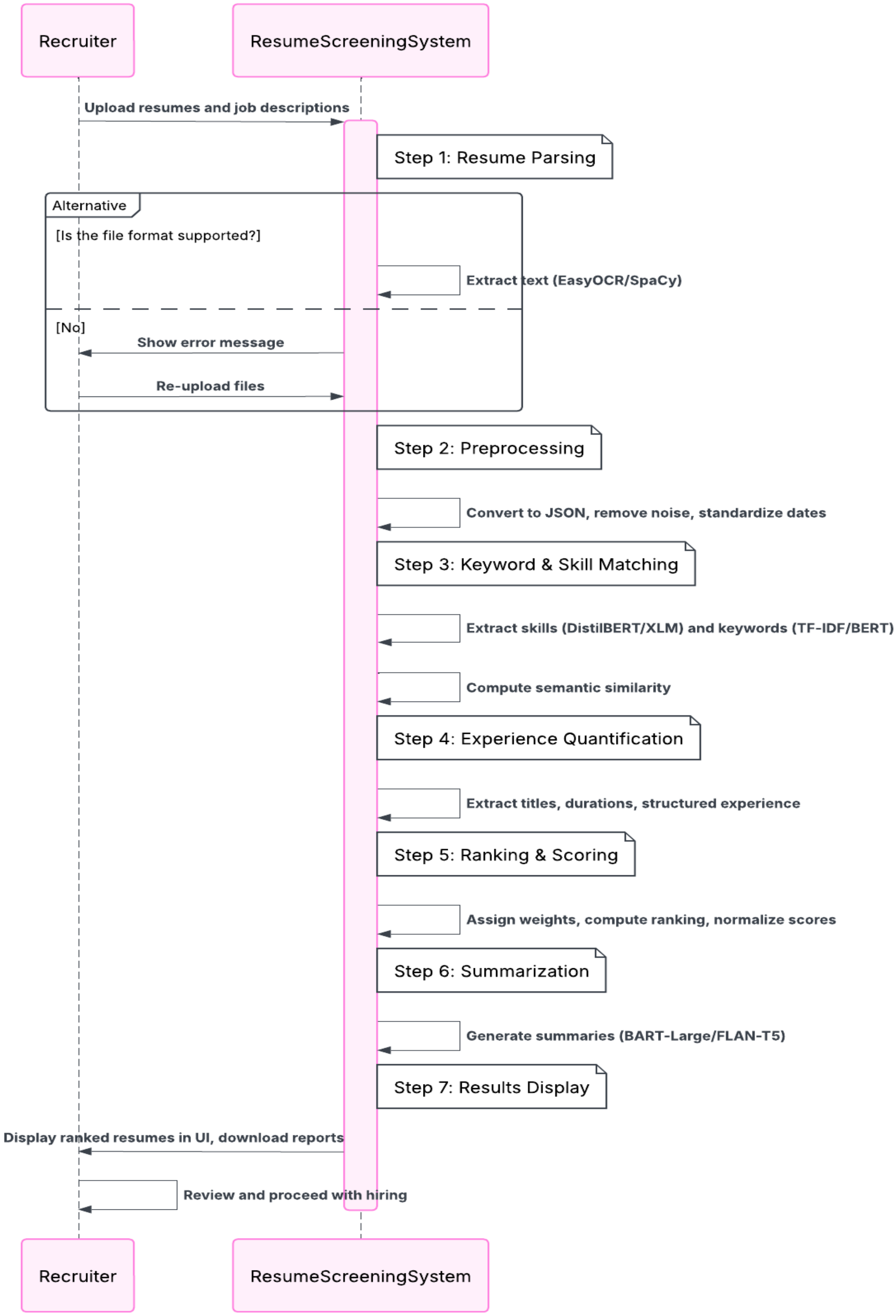
C. Resume Ranking – Cosine Similarity + Weighted Scoring

Cosine similarity measures the similarity between resume embeddings (generated using DistilBERT) and job descriptions. Weighted scoring allows customization, assigning different importance to skills, experience, and education based on recruiter priorities (e.g., skills = 40%, experience = 30%). This approach outperforms simple keyword-based methods (like TF-IDF) by considering the semantic meaning of resume content, resulting in more accurate rankings.

D. Summarization – BART-Large (GPU) / FLAN-T5 (CPU)

BART-Large is a transformer model specialized in abstractive summarization, producing human-like summaries. It requires a GPU for fast processing and high-quality output. FLAN-T5 is a lightweight alternative that runs efficiently on CPU while providing reasonable summarization quality. This combination ensures that summarization can be tailored to available hardware, offering flexibility without compromising accuracy.

Workflow Diagram



Workflow of the Project

Project Workflow Overview:

1. **Resume Dataset Collection:** Collect resumes in PDF, DOCX, and TXT formats along with job descriptions. Convert resumes into JSON using PyMuPDF and python-docx, then clean the text by removing special characters, normalizing case, and applying lemmatization.
2. **Resume Parsing:** Extract text using EasyOCR for image-based PDFs and SpaCy for Named Entity Recognition (NER). Store structured data (name, email, skills, experience, education, certifications) in JSON/CSV format.
3. **Keyword & Skill Matching:** Extract skills using DistilBERT/XLM and match them with job descriptions using TF-IDF or BERT embeddings, computing similarity scores via cosine similarity.
4. **Experience Quantification:** Identify job roles and durations using BERT and regex, then categorize experience into specific skill sets based on predefined thresholds.
5. **Ranking & Scoring System:** Assign weights to skills, experience, and certifications based on job priorities. Calculate final rankings using cosine similarity and weighted scoring, normalizing results for fairness.
6. **Summarization:** Generate candidate summaries using BART-Large (GPU) or FLAN-T5 (CPU), highlighting key details like top skills, experience, and certifications.
7. **Web Application Development:** Build a Flask-based backend with API endpoints for resume processing and ranking retrieval. Create a React/HTML-CSS frontend for recruiters to upload resumes and view results.
8. **Testing & Debugging:** Perform unit and integration testing on all components, and optimize performance using multi-threading to reduce response time.
9. **Deployment & Scaling:** Deploy the backend on Render/Heroku and the frontend on Netlify/Vercel using Docker for containerized deployment, ensuring scalability.
10. **Future Enhancements:** Add multilingual support, integrate with Applicant Tracking Systems (ATS), and improve AI-driven interview question suggestions.