DALHOUSIE UNIVERSITY

CSCI 6509 – Advanced Topics in Natural Language Processing

Submitted by:

Kavya Bhojani (B00981006)

Shrey Monka (B00971102)

Contents

Project Title

Resume Matching Tool

Team Members of the Group

Shrey Devendrabhai Monka – B00971102

Kavya Bhojani – B00981006

Problem Statement

This project aims to build an intelligent system that compares resumes to job descriptions, providing a matching percentage and actionable feedback to help students, job seekers, or employees optimize their resumes for specific roles. The system will extract key information from resumes and job descriptions, analyze similarities and gaps, and suggest improvements to enhance job suitability. This tool will assist users in making their resumes ATS-friendly, improve their job match score, and increase their chances of being shortlisted.

List of possible approaches

1. Text Similarity and Traditional NLP Approaches [1]

- **TF-IDF (Term Frequency-Inverse Document Frequency)**
  TF-IDF transforms text into numerical vectors by assigning higher weights to terms that appear frequently in one document but rarely across the entire corpus. This helps identify important keywords in resumes and job descriptions.
- **Cosine Similarity**
  Cosine similarity measures the angle between two vectors (e.g., resume and job description). A score close to 1 indicates high similarity, while a score close to 0 suggests minimal overlap. Using TF-IDF with cosine similarity provides a straightforward way to assess how well the content of a resume aligns with a job description.
- **Bag-of-Words (BoW)**
  This approach counts word occurrences without considering their order. While it is simple, it can miss contextual relationships between words, making it less effective for nuanced analysis.
- **Limitations**
  These techniques cannot capture the meaning or semantic relationships between words (e.g., "ML" and "Machine Learning" would be treated differently). As a result, they may struggle with complex job descriptions or varied resume formats.

2. Deep Learning and Transformer-Based Approaches [2]

Transformer models have transformed the field of NLP by capturing the contextual meaning of words, phrases, and entire documents. These models excel at understanding the semantic alignment between resumes and job descriptions.

- **BERT (Bidirectional Encoder Representations from Transformers)**
  BERT is a state-of-the-art model that captures bidirectional context, understanding both left and right context simultaneously. Fine-tuning BERT on job descriptions and resumes allows the system to recognize nuanced relationships between qualifications, skills, and job requirements.
- **S-BERT (Sentence-BERT)**
  An extension of BERT, S-BERT provides sentence-level embeddings, which are particularly useful for comparing entire sentences or sections in resumes with job postings. Using cosine similarity on S-BERT embeddings improves matching accuracy by focusing on semantic similarity.
- **RAG (Retrieval-Augmented Generation)**
  RAG combines retrieval-based approaches with generative NLP models. It can retrieve relevant information from job descriptions and generate tailored feedback for the user, enhancing both matching and feedback generation capabilities.
- **Use Case in Matching Systems**
  These models capture relationships such as synonyms, abbreviations, and domain-specific terms (e.g., “ML” and “Machine Learning”), making them ideal for resume matching. They can also identify transferable skills, which traditional methods often miss.

3. Skills Taxonomy and Ontology-Based Methods

Skill taxonomies and ontologies provide a structured way to categorize and relate skills, tools, certifications, and job roles. These methods help bridge the gap between different terminologies and identify transferable skills.

- **Hierarchical Skills Taxonomy**
  A hierarchical taxonomy organizes skills into parent-child relationships (e.g., “Software Development” > “Backend Development” > “Django”). Using this approach allows the system to detect related skills even when they are expressed differently in resumes and job descriptions.
- **Ontology-Based Matching**
  Ontologies represent knowledge as a graph of interconnected entities, capturing the relationships between various skills, job titles, and qualifications. This enables the system to map out latent connections between job requirements and a candidate’s experience.

- **Skill Synonym Mapping**
  Ontology-based methods allow the system to identify different ways of expressing the same concept (e.g., “Data Analysis” vs. “Data Science”). This improves matching accuracy by aligning the terminology used in resumes with the language in job descriptions.
- **Use Case**
  These methods are especially useful for identifying transferable skills, such as recognizing that experience with Docker could partially fulfill a requirement for Kubernetes expertise.

4. Hybrid Matching Approaches

A combination of traditional NLP techniques, deep learning, and taxonomy-based methods can further enhance matching accuracy.

- **Feature Engineering with TF-IDF + BERT Embeddings:**
  Combining TF-IDF scores with deep learning embeddings captures both term-level and contextual-level similarities. This approach leverages the best of both worlds: fast calculations with TF-IDF and nuanced understanding with BERT.
- **Knowledge-Driven Systems with NER (Named Entity Recognition)**
  Incorporating NER techniques from spaCy or BERT can extract key entities (skills, certifications, and job roles) from resumes and job descriptions. These entities can then be compared and analyzed using cosine similarity or graph-based techniques.
- **Use Case for Feedback Generation**
  Hybrid approaches allow for more precise feedback. For example, missing entities detected by NER can be highlighted, and deep learning models can provide suggestions on how to phrase new additions effectively.

Project plan for the rest of the term

Week 1 (Oct 28 - Nov 3): Set up the project environment using React and Vite, design the component architecture, and implement basic UI elements like file upload and job description input. Deliverables: Working frontend with basic upload functionality and a component diagram.

Week 2 (Nov 4 - Nov 10): Enhance the UI with features like drag-and-drop, file validation, and responsiveness. Begin backend setup by designing the API structure and endpoints. Deliverables: Complete frontend, functional file upload, and basic API structure.

Week 3 (Nov 11 - Nov 17): Implement core features such as resume, and job description parsing and integrate an NLP model (BERT/RAG) for matching with similarity scoring. Deliverables: Working parsers and an initial matching system.

Week 4 (Nov 18 - Nov 24): Add advanced features like skill gap analysis, experience matching, and feedback generation. Develop dashboard components for visualization. Deliverables: Complete matching system, feedback generator, and interactive dashboard.

Week 5 (Nov 25 - Dec 1): Perform system integration, optimization, bug fixing, and user testing. Finalize documentation and prepare the application for deployment. Deliverables: Fully integrated, deployment-ready system with complete documentation and final report.

References

[1] "IJERT-Resume Classification and Ranking using KNN and Cosine Similarity," [Online]. Available:https://www.academia.edu/51588484/IJERT_Resume_Classification_and_Ranking_using_KNN_and_Cosine_Similarity?sm=b. [Accessed 27 October 2024].

[2] "Talent recommendation based on attentive deep neural network and implicit relationships of resumes," [Online]. Available:https://www.sciencedirect.com/science/article/pii/S0306457323000948. [Accessed 27 October 2024].